



BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

VOLUMETRIC SEGMENTATION OF DENTAL CT DATA

SEGMENTACE ZUBNÍCH OBJEMOVÝCH DAT

BACHELOR'S THESIS

BAKALÁŘSKÁ PRÁCE

AUTHOR

AUTOR PRÁCE

MATEJ BEREZNÝ

SUPERVISOR

VEDOUCÍ PRÁCE

Doc. Ing. MARTIN ČADÍK, Ph.D.

BRNO 2020

Zadání bakalářské práce



Student: **Berezný Matej**
Program: Informační technologie
Název: **Segmentace zubních objemových dat**
Volumetric Segmentation of Dental CT Data

Kategorie: Počítačová grafika

Zadání:

1. Seznamte se s metodami pro segmentaci objemových dat. Pozornost věnujte automatickým, poloautomatickým i plně manuálním metodám.
2. Vytipujte metody vhodné pro segmentaci dentálních objemových dat z počítačového tomografu (CT). Popište vlastnosti vybrané metody.
3. Na základě vybrané metody navrhnete a implementujete systém pro segmentaci zubních objemových dat.
4. S navrženým systémem experimentujte a vyhodnoťte dosažené výsledky.

Literatura:

1. Dle pokynů vedoucího.
2. Data dodá vedoucí práce.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz <https://www.fit.vut.cz/study/theses/>

Vedoucí práce: **Čadík Martin, doc. Ing., Ph.D.**

Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.

Datum zadání: 1. listopadu 2020

Datum odevzdání: 12. května 2021

Datum schválení: 30. října 2020

Abstract

The main goal of this work was to use neural networks for volumetric segmentation of dental CBCT data. As a byproducts, both new dataset including sparse and dense annotations and automatic preprocessing pipeline were produced. Additionally, the possibility of applying transfer learning and multi-phase training in order to improve segmentation results was tested. From the various tests that were carried out, conclusion can be drawn that both multi-phase training and transfer learning showed substantial improvement in dice score for both sparse and dense annotations compared to the baseline method.

Abstrakt

Hlavným cieľom tejto práce bola segmentácia objemových CT dát za použitia neurónových sietí. Ako vedľajší produkt bol vytvorený nový dataset spolu s silnými aj slabými anotáciami a nástroj pre automatický preprocessing dát. Takisto bola overená možnosť využitia transfer learningu a viacfázového tréovania. Z mnohých vykonaných testov možno vyvodiť záver, že aj tranfer learning aj viacfázové tréovanie mali pozitívny vplyv na vývoj dice skóre v porovnaní so základnou použitou metódou či už pri silných, alebo slabých anotáciách.

Keywords

Image processing, segmentation, volumetric segmentation, U-Net, CT scans, CBCT scans, medical data, deep learning, convolutional neural networks, sparse annotations, dense annotations, transfer learning, multi-phase training, image restoration

Klíčová slova

Zpracovanie obrazu, segmentácia, objemová segmentácia, U-Net, CT skeny, CBCT skeny, medicínske dáta, hlboké učenie, konvolučné neurónové siete, slabé anotácie, silné anotácie, transfer learning, viacfázové tréovanie, obnova obrazu

Reference

BEREZNÝ, Matej. *Volumetric Segmentation of Dental CT Data*. Brno, 2020. Bachelor's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Doc. Ing. Martin Čadík, Ph.D.

Rozšířený abstrakt

Počítačová tomografia (CT) sa stala od svojho vzniku v sedemdesiatych rokoch minulého storočia neoddeliteľnou súčasťou modernej medicíny. Slúži na diagnostiku širokého spektra poranení či chorôb a jej základným princípom je zber a počítačové spracovanie veľkého množstva údajov o hodnote absorpcie röntgenového žiarenia. Výsledkom počítačového spracovania sú 3D objemové CT dáta reprezentujúce určitú časť ľudského tela. Samotnú počítačovú tomografiu je možné rozdeliť na rôzne typy, pričom jedným z najnovších typov je CT využívajúca kužeľovitého zväzku lúčov (CBCT), ktorý sa najčastejšie používa v zubnom lekárstve.

Účelom tejto práce bola segmentácia zubných koreňov a koruniek z takýchto CBCT objemových dát. Z rozsiahleho spektra segmentačných metód, ktoré sa v praxi bežne používajú, bolo vybrané hlboké učenie ako metóda pre automatickú segmentáciu. Konkrétne sa jednalo o konvolučné neurónové siete (CNN), ktoré momentálne dosahujú omnoho lepšie výsledky než iné segmentačné metódy. Čo sa týka samotnej architektúry konvolučnej neurónovej siete, vzhľadom na povahu cieľových dát bola zvolená architektúra 3D U-Net.

Momentálne neexistujú žiadne voľne dostupné zubné datasety, ani anotácie potrebné pre tréning takejto neurónovej siete, preto bolo nutné získať tréningový dataset iným spôsobom a dodatočne k nemu vyrobiť anotácie. Tréningový dataset bol napokon zložený z 42 CT skenov zubných oblastí extrahovaných z veľkého datasetu 500 ľudských hláv a štyroch ďalších CBCT skenov voľne dostupných na internete. K tomuto datasetu boli ručne vytvorené dva typy anotácií, a to slabé a silné anotácie.

Vzhľadom na to, že tréningové dáta pochádzali z rôznych zdrojov a mali rozdielny formát, bolo ich nutné pred vstupom do neurónovej siete dodatočne spracovať. Spracovanie prebiehalo v štyroch krokoch - prevzorkovanie, normalizácia, orezávanie a vyplnenie.

Všetky tréningové dáta boli prevzorkované na izotropické rozostupy voxelov o veľkosti 0.8mm. Cieľom tohoto prevzorkovania je dosiahnuť rovnaké dimenzie pre oba typy skenov (CT a CBCT) a zároveň zmenšenie ich celkovej veľkosti pre urýchlenie tréningu. Ďalším krokom bolo škálovanie dát do rozsahu od 0 do 1, orezanie zubnej oblasti na základe dopočítanej masky a pridanie dostatočného množstva výplne tak, aby koincidovali rozmery obrázku so vstupom akceptovaným neurónovou sieťou. Nakoniec bola na všetky dáta aplikovaná z-score normalizácia.

Čo sa týka samotnej stratégie tréningu, okrem konvenčného spôsobu bola pridaná aplikácia transfer learningu, viacfázové tréningovanie a augmentácia dát. Pri použití transfer learningu boli modely určené pre segmentáciu predom natréňované na účel obnovy obrazu buďto z veľkého množstva CT skenov hrudníka, alebo malého množstva zubných CBCT skenov. Viacfázové tréningovanie pozostávalo z prvotného tréningu na neorezaných skenoch, následne na orezaných skenoch a nakoniec na vybranej vzorke orezaných CBCT skenov. Jednotlivé techniky slúžiace na augmentáciu dát boli následovné: náhodné rotácie, náhodné pretočenia, elastické deformácie, augmentácia za pomoci gamma korekcie a náhodná zmena kontrastu.

V prvej časti experimentov bolo testovaných niekoľko parametrov tréningu za účelom dosiahnutia čo najlepších budúcich výsledkov. Na základe nameraných výsledkov bola zvolená optimálna veľkosť vstupu do CNN 64x64x64, rýchlosť učenia bola nastavená na 10^{-3} v prípade silných anotácií a 10^{-4} v prípade slabých anotácií. Takisto bolo dokázané, že aplikovanie z-score normalizácie zvýšilo segmentačnú presnosť o zhruba 5%.

Ďalšia sada experimentov testovala výsledky jednotlivých prístupov zmienených v predošlých odstavcoch pri jednofázovom tréningu. V oboch prípadoch bolo možné pozorovať dominanciu transfer learningu nad "čistými" modelmi.

Posledná sada experimentov bola zameraná na zvýraznenie vplyvu viacfázového tréningu na celkovú segmentačnú presnosť. Z výsledkov možno vydedukovať, že celkové zvýšenie presnosti oproti jednofázovému tréningu sa pohybovalo okolo 10% pre slabé anotácie a 5% pre silné anotácie. Takisto si je možné všimnúť, že model predom natrénovaný na obnovu obrazu z CT skenov hrudníku svojimi výsledkami v oboch prípadoch predčil všetky ostatné modely.

V konečnom dôsledku bolo vytvorených niekoľko segmentačných modelov, pričom najlepší z nich dosiahol úctyhodných výsledkov na testovacej sade - 90.23%, 87.2% a 91.70%.

Volumetric Segmentation of Dental CT Data

Declaration

I hereby declare that this Bachelor's thesis was prepared as an original work by the author under the supervision of Mr. Doc. Ing. Martin Čadík Ph.D. The supplementary information was provided by Mr. Ing. Oldřich Kodým. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....

Matej Berezný

May 7, 2021

Acknowledgements

I would like to thank my supervisor Ing. Martin Čadík Ph.D. for providing valuable insights to the problematic, everlasting motivation and professional guidance. Additionally i would like to thank my family and friends for supporting me throughout the process of writing this thesis. Heartfelt thanks belongs also to Metacentrum organisation, for allowing access to their resources used for convolutional neural network training.

Contents

1	Introduction	2
2	Image segmentation	4
2.1	Medical images	4
2.2	Manual segmentation	5
2.3	Semi-automatic segmentation	6
2.4	Graph Cut Algorithm	6
2.5	Level-set methods	8
2.6	Automatic segmentation	10
2.7	Deep learning in medical imaging	11
3	Methodology	14
3.1	Prior problems	14
3.2	Datasets	15
3.3	Training strategy	18
3.4	Preprocessing	21
4	Implementation	26
4.1	Data manipulation	26
4.2	Training and validation	27
4.3	Launching scripts	28
5	Experiments	29
5.1	Determining the best training parameters	29
5.2	Single-phase training	32
5.3	Multi-phase training	35
6	Conclusion	39
	Bibliography	40
A	Contents of the included storage media	45
B	Dataset	46

Chapter 1

Introduction

Since its discovery in 1970s, Computed Tomography became inseparable part of the modern medicine. Along with Magnetic resonance imaging (MRI) and their predecessor, X-ray, they are being classified as medical imaging techniques, since their outcome represents either 2D or 3D images. In medicine, their purpose lies within preventive healthcare or as a tool used in screening for disease. Preventive healthcare, as the name might suggest, focuses on a prevention of illness to decrease the burden of disease and associated risk factors. Although at first glance, these techniques might seem interchangeable between each other, they do differ in multiple areas.

Principle of magnetic resonance imaging lies within special devices, called MRI scanners, generating strong magnetic fields and radio waves that bounce off the fat and water molecules in human body. Radio waves are being transmitted to the receiver located in the machine which further translates them into an 3D images of the organs in the body used in medical diagnosis. Some patients with medical implants or other non-removable metal inside their body may also be unable to undergo the examination, due to use of before-mentioned strong magnetic fields. Images produced by technique of magnetic resonance imaging typically achieve better results in intricate soft tissue visualization compared to both X-ray and CT, however that comes at the cost of increased price. Therefore, MRI is more suitable for examining soft tissue injuries, especially in tendons and ligaments, or discovering brain tumors and spinal cord injuries.

X-rays, similarly to visible light, belong to the category of electromagnetic rays, so they follow the rules of electromagnetic radiation. Radiograph, the outcome of the x-ray procedure, is made by electromagnetic beams produced from x-ray source passing through the human body. X-rays traveling through the human body get absorbed in different amounts determined by the radiological density of the given tissue. Various weakened x-rays are then being recorded by the x-ray detector and sent to the machine to produce 2D scan of the patients body.

Computed Tomography, or more commonly known as CT, can be viewed as more powerful, more advanced form of the x-ray procedure. It uses high performance x-ray lamp in combination with multiple rows of detectors, which quickly rotate around the patient, producing signals that are further being processed by a computer to produce cross-sectional

images also called „slices“. These slices can be later projected into multiple planes, and further processed to the three-dimensional images.

Medical imaging techniques, predominantly x-rays and CTs, are not foreign to the dentistry either, since their use can be often observed in dental implant placements, treating jaw tumors, reconstructive surgeries etc. While its not entirely uncommon to see the use of MRI as well, for now it falls behind computed tomography due to its lesser availability, higher price and often difficulties with metallic artifacts.

Although analysing and evaluating single x-ray scan might not be a difficult procedure, analysing enormous amounts of raw data produced by computed tomography in short time period may prove to be near impossible task even for experienced dental radiologist. To shorten the time spent on analyzing such data, decrease occurrences of oversights and reduce the number of false negatives, computer aided detection (CAD) was introduced. Although it is impossible for computer aided detection to completely replace professionals, it allows them to only control results of the computer’s work instead of devoting their precious time.

Computer aided detection is a term describing systems capable of processing raw computed tomography data, further transforming them according to the specific needs of a doctor. It combines features from artificial intelligence and computer vision with image processing. Most frequent uses represent detection and highlighting of suspicious objects such as tumors and transformation of two-dimensional slices into three-dimensional models needed for surgery planning. There has been a major breakthrough in computer aided detection relatively recently, due to the introduction and advancement of convolutional neural networks. With their use, more accurate results of segmentation and classification can be achieved, often even shortening the amount of time needed.

The purpose of this thesis will be to develop a system suitable for producing 3D segmentations of teeth without the surrounding mandible and maxilla bones or any other types of soft tissue using convolutional neural networks in combination with other known image processing algorithms.

Chapter 2

Image segmentation

The term 'Image Segmentation' defines collection of methods, that firmly belong to the field of computer vision. The basic principle of segmentation methods lays in localization and visual separation of pixel/voxel groups, depending on the number of dimensions in given digital image, that satisfy certain criteria set by the user or algorithm. The name itself comes precisely from these groups of pixels/voxels, since they are often being called **segments**. Outcome of such procedure is, in most cases either set of labeled segments, that covers entirety of the image area, or set of contours, extracted from the original image [7]. The main purpose of image segmentation is to pinpoint and label certain important objects, such as people or animals, hence simplifying the image for further processing. Techniques used for labeling these objects can be further divided into two separate styles. First style, called **semantic segmentation** specializes in assigning the same label to the each and every object, that is sharing similar key characteristics (shape, color etc.), therefore having the same semantic significance. In addition to performing exactly the same task as first style, the second one, named **instance segmentation**, manages to individually segment objects, that would, with the use of semantic segmentation, belong to the same category (instance). Example of the semantic segmentation could be segmenting tooth-like objects from the dental scan, while instance segmentation would be taking it a step further, assigning specialized label to every single one of them.

In modern radiology, different approaches to the segmentation of medical images can be observed. To summarize them into the three distinct categories by their method of execution and requirement for human interaction, scientists describe them as **manual**, **semi-automatic** and fully **automatic segmentation**. Even though most of the methods belonging to these three categories are often used separately from each other, their combination may result in higher performance and segmentation quality.

2.1 Medical images

Raw data for 3D medical segmentation come in form of computed tomography scans. Each scan consists of exact amount of **slices**, 2D images representing 'slices' or cuts of the human body usually in horizontal, coronal or sagittal plane. Slices contains various information

about themselves, such as slice thickness determining the scan resolution and slice increment, that refers to the distance between two neighbouring slices. Raw slices are usually saved in either dicom format (.dcm) as individual slices, or in Nifty image format (.nii), representing entire patient volumes.

Large collection of such data from many patients is called **dataset**. Datasets may or may not contain **labels**, image masks that correspond with certain objects (usually organs) located in slices. These labels are referred as **ground truth** or 'gold standard' and are often manually segmented by radiologists. They are needed for validation of machine learning algorithms such as neural networks, either in supervised training, or inference. Efficiency of such machine learning algorithms is determined by **dice similarity coefficient**, statistic that calculates similarity between two objects, for example segmentation outputs and their corresponding ground truth labels.

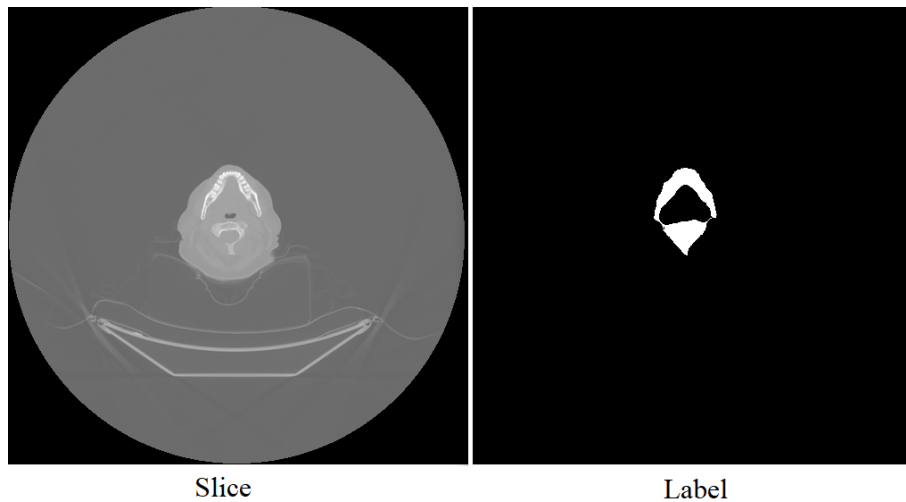


Figure 2.1: Example of slice and its respective annotation

2.2 Manual segmentation

Manual segmentation and classification is the oldest and probably still most widely spread method for segmenting medical data to this day. The reasons for its continuous popularity among doctors are no need for higher education in the field of information technologies, relatively low hardware requirements and general execution simplicity, making the technique learning process fairly short. All of these aforementioned advantages come at the cost of major drawbacks, such as inconsistencies in segmentation appearing due to the human factor (for example exhaustion) or frequent appearance of image distorting metallic artifacts, oversights of the important disease indicators, occurrences of false positives as a consequence of lowered image quality being produced by cheaper computed tomography scanners. Last, but certainly not least important drawback of the manual segmentation is its time consumption, since it requires for radiologist to segment every slice manually, while trying to stay as precise as possible.

The usual way manual image segmentation is performed, is with the help of the third party programs, so called image analysis tools, such as ITK-SNAP, 3D Slicer, MedSeg, et cetera. First step usually consists of creating labels with specific color for each area of interest, for example different colored labels for each type of teeth, mandible and maxilla bone. Next, radiologist locates these areas of interest on every slice, and 'colors' them accordingly to their labels. Coloring is done on similar basis as in other commercial graphic editors, typically either with paint brush or with use of polygonal contours. After finishing the initial segmentation, radiologist may choose to completely get rid of, or at least reduce inconsistencies in between equally labeled areas on different slices. Inconsistencies usually represent significant differences in sizes of identically labeled areas, or accidental uses of unwanted labeling on certain areas. Not resolving these inconsistencies may result in severe inaccuracy of the final result of segmentation. Lastly, various image analysis tools allow displaying results of the work as 3D model, visualizing only segmented areas from slices.

Perspective usage of manual data segmentation seems to be in the development of more complex, semi-automatic or even fully automatic segmentation methods. Clinical data segmented manually by experienced radiologist often serve as a gold standard/ground truth for testing, validation and quality assessment for above-mentioned methods [35].

2.3 Semi-automatic segmentation

The problem of extensive time consumption required by manual segmentation and general coarseness of automatic segmentation algorithms before the upsurge of convolutional neural networks led scientists to discovery of newer, more acceptable solutions. These solutions were later categorized as semi-automatic (also known as interactive) segmentation methods. They provide promising compromise between previously mentioned segmentation methods, combining processing speed of automatic algorithms and precision provided by human touch. The human factor usually plays important role in either pre or post-processing, where doctors supply the critical high-level instructions that are essential for guaranteeing appropriate behaviour of interactive algorithms. Such high-level instructions typically include setting the starting point of the algorithm, or marking areas of the image as foreground or background [22]. Rest of the segmentation procedure is handled by these special interactive algorithms.

In recent years, popularity of advanced interactive segmentation methods based on using **graph cuts** or **level-set methods** increased substantially, mainly due their ability to handle even more complex segmentation tasks, such as volumetric processing brain or dental datasets.

2.4 Graph Cut Algorithm

The basis, on which the Graph cut algorithm is built upon originates from one of the mathematical fields, named graph theory. There, if the partition of connected graph into specific disjoint subsets is desired, **cut** is performed. Severance of the graph's connectivity

can be achieved with removal of various elements, such as cut vertices, cut edges or cut sets [28]. If elimination of the single edge causes graph to be disconnected, that edge is called cut edge, where the same principle applies for vertices as well. Cut set can be defined as an array of edges, which deletion would cause desired effect. Every cut is defined by its weight, representing number of edges crossing the cut in unweighted graph or total sum of edge weights in weighted graphs. Cuts are further divided into subcategories determined by their weight and purpose. The simplest of them are minimum cut, which seeks to find minimal weight required for performing the cut and maximum cut[3], which does the exact opposite.

The graph cut algorithm uses flow-networks or directed weighted graphs $G = (N, E)$, where N represents set of both terminal and non-terminal nodes and E set of directed edges connecting given nodes. Terminal nodes located in N are called source s and sink t . Non-terminal nodes usually portray voxels or pixels, and terminal nodes may for example depict labels, or simulate foreground and background. Edges E consist of n-link and t-link edges, where directed n-link edges usually link pairs of neighbouring pixels or voxels $\{p, q\}$ and have assigned non-negative weight, also called flow $w(p, q)$, that might differ from reverse n-link edge $w(q, p)$. T-link edges connect pixels with terminal nodes, and their weight represents penalty for assigning specific terminal label [26].

In computer vision, graph cuts had proven to be effective tool for solving multitude of fundamental tasks, such as image smoothing and segmentation or finding the corresponding sets of points between two images in the same 3D scene. In graph cut algorithm, N-dimensional images can be represented as graphs, and segmentation process as graph cutting, where, with the optional help of the user input, algorithm divides graph into different subsets separating background and foreground, which symbolizes desired object, by solving **max-flow min-cut problem**. To accomplish this source-sink or s-t cut on such graph, terminal nodes are required to be in different subsets. User input in graph cut algorithm usually refers to placing initial restrictions upon image by manually assigning object label to pixels and pixel areas, where the user is absolutely certain about them belonging into either background or foreground. Example of such user input could be indicating focus areas for each tooth in dental dataset segmentation.

The point of the max-flow min-cut theorem [2] is to find optimal way to perform s-t cut in a flow network. The theorem can be further divided into two subproblems, and that finding max-flow and min-cut on the flow-network. **Max-flow** can be described as maximum flow rate that is allowed to pass directly from source whilst finishing in sink. Loose interpretation of max-flow could be the maximum amount of water (flow rate), that can pass through pipes (edges) from water source to the sink. **Minimum-cut**, in this case, is trying to achieve minimal flow, or total sum of the weighted edges that would cross the s-t cut separating flow network into the two subsets, each with their terminal node. According to the theory presented by Ford and Fulkerson [6], max-flow of the given network is equal to the minimal cut weight separating s and t , therefore resolving only one of subproblems leads to the complete solution.

Application of graph cut algorithm, typically in combination with other techniques is not foreign to the segmentation of clinical dental data either. As presented in study by T. Evain et al [5], use of the graph-cut in connection with statistical shape analysis and user interaction achieved almost state of the art results, with average Dice coefficient above **0.95**.

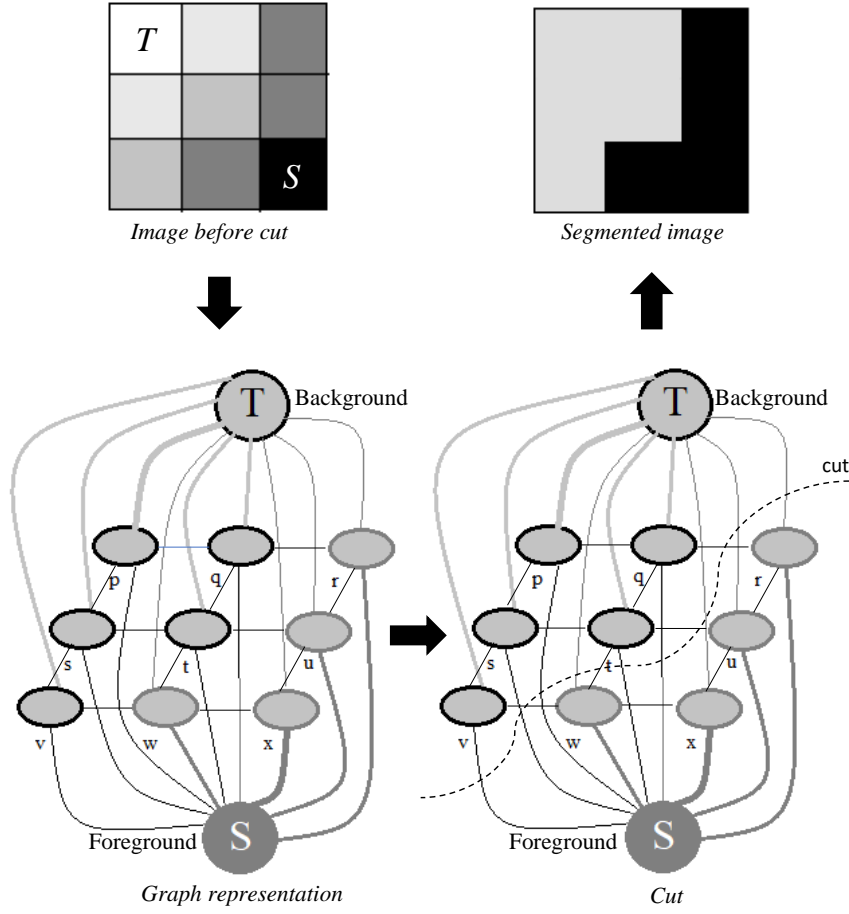


Figure 2.2: Illustration of a graph cut

Even though high Dice coefficient was achieved, authors themselves reported appearance of false edges in presence of metallic artefacts, that might severely impact segmentation results.

Another example of integration of graph-cut based approach into segmentation of dental datasets is in pairing with Hidden Markov fields [12]. By using 3D Hidden Markov fields as tool for interpreting 3D volumetric data, solid segmentation results were achieved with graph-cuts. Authors achieved average dice similarity coefficient of **0.89**, while attempting to segment individual teeth from dental scans. No training or testing datasets including metallic artifacts were documented, so no definite conclusion can be drawn about their effect on accuracy of the results.

2.5 Level-set methods

For many years, level-set methods became most studied and widely used methods for medical image segmentation. They can be regarded as an effective conceptual framework utilizing level-sets for numerical representation of contours and surfaces. Their advantage lies

in portraying such objects implicitly as functions in Cartesian grid, without any need to explicitly parameterize them [32].

Initial thought behind introducing level-sets to image segmentation was to process constantly changing contours and surfaces of desired objects, with occurrence of their topology changes, development of sharp edges and peaks. Its not uncommon that the initial starting point of level-set method is vastly different from its point of convergence in the final result, since contours and surfaces can merge together, split apart or develop holes in the process. This modality of level-set based approach allows it to be effective in processing objects without any prior knowledge about their shape.

Largest obstacle in development of such level-set method is definition of the right **speed function**. Speed function is used for determining the new position of each point on curve/-surface when level-set method reaches next state. The most common speed functions are typically based on gradients, edge strength and region density, which are sufficient for solving less complex segmentation problems, but might sometimes prove unsatisfactory for segmentation of difficult clinical data.

Typical example of aforementioned hard-to segment data are teeth datasets, since facial region is known to be one of the most frequent regions for the occurrence of foreign particles in form of dental implants. Dental implants, also known as fixtures are used to fixate dental prostheses such as crowns, bridges and dentures to the mandibular/maxillary bone. And since material used for creation of such fixture is, in the most cases, of metallic origin, teeth projection in the CT scans often differs from reality. That may lead to unwanted changes in shape of surface / curve representing tooth. Secondly, crown areas of the individual neighbouring teeth may touch each other, therefore making the localization of teeth boundaries much harder. Another problem lies in occurring image noise and similar pixel intensities between root area of the teeth and alveolar bone surrounding it.

One of the possible approaches on how to tackle these problems might be with the use of advanced form of level-set methods called **hybrid level-set model** [8]. Hybrid level-set model is segmentation method that combines three different variations of basic level-set methods used in medical segmentation with addition of user input in form of selection of starting segmentation slice that will set the direction of level set methods. Previously mentioned three level-set methods used in hybrid model are **edge-based** models, **global region-based** models and **local region-based** models. Edge-based models are efficient in detection of object boundaries, therefore stopping the curve evolution, but fall short in presence of image noise and weak edges that occur exactly when the neighbouring crowns touch each other. Principle of global region-based models lies in approximation of statistical pixel intensities on whole image plane, hence separating image to the background and foreground. That increases the overall tolerance to image noise and initial conditions, but suffers greatly from similar neighbouring intensity statistics such as alveolar bone and tooth roots. Last method operates on the similar principle as previous one, but instead of focusing on global intensity statistics it uses local intensity with reliance on initial condition, solving both problem with weak edges and problem with similar neighbouring intensities. Every single one of these methods has its own drawbacks, but when used in conjunction, they do complement each other fairly well.

Even though this method solves many complications linked with the segmentation of dental computed tomography images, while still reaching higher average dice than previous attempts on teeth segmentation using level-sets, it imposes specific requirements on patients to stay in open-bite position while being scanned, since closed bite would result in contour connection between upper and lower teeth line. Also, the image degradation caused by metallic artifacts such as dental implants proves to be limiter even for current version of hybrid level-set method.

2.6 Automatic segmentation

Thresholding

Thresholding could be considered as probably the simplest automatic segmentation method that exists. It performs a binary segmentation of an original image into an image with only two colors (usually black and white) based on specific criterion called threshold, that is set beforehand. Image intensity contained in each pixel from original image is compared to the threshold and based on the result of the comparison gets assigned its new binary value. Thresholding is usually performed on images with single channel, for example images converted to grayscale, but its principle could be applied to any arbitrary number of channels, as long as sufficient thresholds are supplied.

The use of thresholding as primary segmentation method in medicine is limited, since any image noise, overlap of multiple different tissue intensities in one pixel or presence of metallic artifacts may completely invalidate the result of segmentation. Its more commonly used as a complementary method, for example to separate specific label from ground truth images with multiple labels, in input image pre-processing or as a part of more sophisticated methods.

Snakes

Snakes, also known as active contour model could be defined as one of the many tools in computer vision to detect and segment objects, whose approximate boundary region is already known. It functions on opposite basis compared to level-set methods, since snakes are **parametric** representations of curves instead of their representation as functions on the Cartesian grid. The segmentation process with snakes itself is iterative, where initial contour/s move closer to the desired object boundaries in each iteration, until they tightly wrap around the object boundaries, hence the name „Snakes“. The strength of such method lies in fast adaptation to differences between input images, finding objects in noisy images or in creation of missing boundaries in incomplete objects. To compute new position of contours in every iteration, active contour model uses **energy function**.

The evolution of active contours in each step is done by reducing and possibly minimizing the „energy“ required to maintain contour in each step. Energy of the curve can be calculated via energy function, consisting of its **internal** and **external** components. Internal energy component focuses only on the curve shape (smoothness) regardless of the

original image, while external energy component calculates how well does the contour fit object boundaries. To achieve the most accurate results, parameter representing importance factor can be placed upon both of these components.

Except for its frequent use in graphic editors such as Adobe Photoshop, snakes are extremely popular in medical imaging as well due to their applicability on often noisy and generally low quality images produced by cheaper CT scanners [21].

2.7 Deep learning in medical imaging

The original attempt at using deep learning for image classification and segmentation was the concept of artificial neural networks. Concept of artificial neural networks (ANN), as one of the machine learning algorithms, was inspired by the central nervous system in human bodies, by connecting layers with artificial neurons [17]. But due to lack of learning data, insufficient computing power artificial neural networks often suffered from overfitting and vanishing gradient problems [13]. These problems resulted in scientists prioritising other, more reliable algorithms for solving segmentation tasks.

But relatively recently, computer hardware advancements and availability of larger datasets made a substantial contribution to deep learning in a form of convolutional neural networks achieving groundbreaking results in variety of fields, including image segmentation and classification, where it broke long-standing all-time records [36].

Convolutional neural networks

Convolutional neural networks (CNNs) share many similarities with ANNs [37], including components, such as neurons, synapses, weights, biases and functions. Main difference between them is that convolutional neural networks are spatially invariant, so there is no need to pay extra attention on location of desired segmentation objects. As the name „convolutional“ implies, CNNs take advantage of an mathematical operation called convolution instead of general matrix multiplication, at least in some of the layers [10]. CNNs have been applied to huge variety of different tasks, with some notable ones being image classification and segmentation, noise reduction, quality improvement or generating new data. Some of the most common architectures include ResNet[11], VGG[31], GoogleNet[33] and finally U-Net[27].

U-Net Architecture

Architecture of general convolutional neural network can be described as stack of hidden layers with addition of input and output layer, that gradually transform input image into an desired output. In case of fully convolutional network architecture U-Net, hidden layers are sorted into contractive path, or **encoder** that is meant to capture the context of the image and relatively symmetrical expansive path, or **decoder** ensuring the its precise localization. In order to localize, high resolution features coming from contracting path are combined

with the upsampled output from appropriate level of expansive path. Followed by successive convolution layer, network can then construct more accurate output based on aforesaid information. While different architectures work with different types of hidden layers, those of generic unet include **pooling** layers in contractive path, **de-convolution** or up-sampling layers in expansive path and **convolutional** layers in both paths.

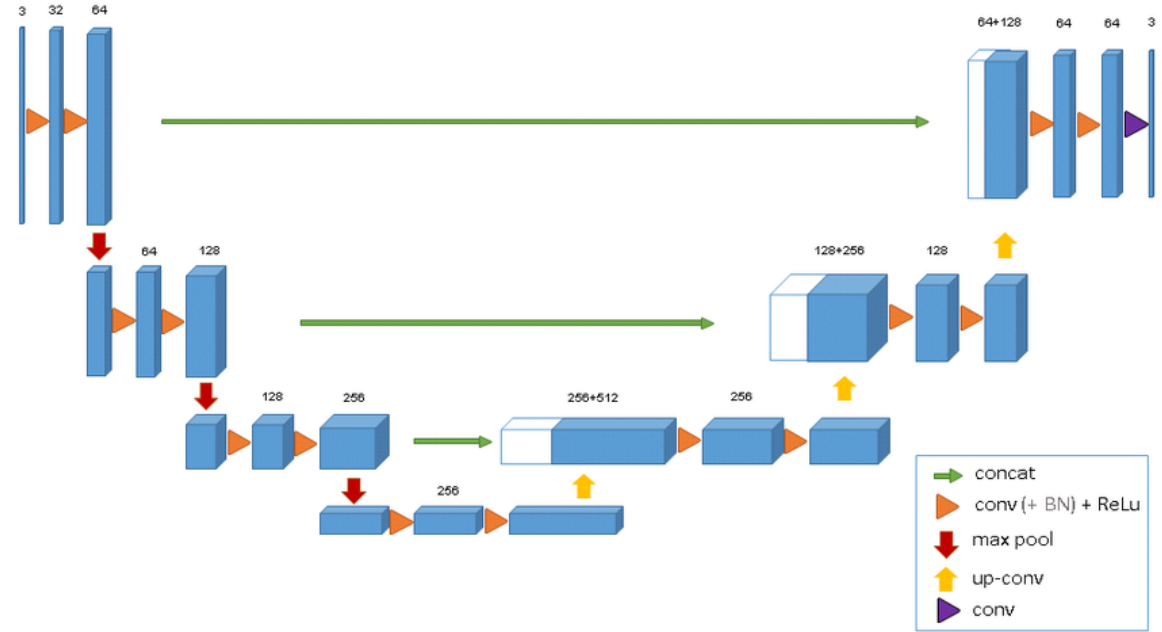


Figure 2.3: Generic 3D U-Net convolutional neural network architecture meant for volumetric segmentation. Contractive path in left side of the model, expansive in the right. Source [42]

Convolution layer, as the name implies, is one of the most important building blocks of CNN. Key characteristic of convolution layer is a set of learnable filters (or kernels) of much smaller size and usually matching number of dimensions compared to the input volume. During the forward pass, each kernel is convolved across every dimension of input, whilst computing the dot product on them, resulting in n-dimensional activation map of each given filter. This allows network to learn such types of filters that will activate upon detection of features specific to target task of neural network at any spatial position in the input [9].

The main function of **Pooling layers** is to lessen the number of feature map dimensions generated by outputs of neuron clusters in previous layer into single neuron in next one. The reason for such layers is to reduce the time and computing power required for such network and to create invariance to small shifts and distortions [40]. There are usually two types of pooling layers, respectively local pooling layers that combine small groups of clusters and global pooling layers, that combine all neurons on corresponding layer. Most popular pooling operations include:

- Max pooling - extracts only the highest value from each of cluster neurons at previous layer

- Average pooling - computes average value from each of the cluster neurons at previous layer

Deconvolution layers, also known as transposed convolution layers are one of the options for upsampling, or decompressing and reconstructing the abstract representations of volumes (feature maps) after passing through contractive path into more desirable outcome. Contrary to other up-sampling techniques such as bilinear or bicubic interpolation, transposed convolution layer allows for interpolation without losing sense of detail, since it uses set of weights learnt during the training process instead of surrounding pixel values.

Learning

Training of neural networks and also many other machine learning algorithms can be approached multitude of different ways depending on the task at hand, but most common concepts are supervised learning, unsupervised learning and semi-supervised learning.

The use of the **supervised learning** can be observed in most of the machine learning algorithms. To be able to train neural network under a supervision, training dataset has to contain two different types of information. Firstly, it contains data that could be considered as input into the CNN, and additionally, the full set of data (also called **dense annotations**) that could be interpreted as an ideal outcome from the neural network architecture. The goal then is, with use of CNN, to approximate function mapping inputs to outputs as precisely as possible, for achieving high accuracy in actual field of use. Method is called supervised because algorithm attempts to make iterative predictions under supervision of a „teacher“, who corrects the mistakes made by algorithm based on supplied labeled data.

Unsupervised learning does not require data to also have their respective annotations prepared, thus neural network is handed dataset without any prior instructions about desired outcome or correct answer. Model then attempts to automatically learn the different patterns and structures present in training data.

Semi-supervised learning is, as its name might suggest, combination of both approaches mentioned above. Typical training datasets used along with semi-supervised learning involve both labeled and unlabeled data. This method is especially useful, when extraction of relevant information from the data proves to be difficult task, or creating dense annotations on entirety of the training dataset consumes too much time. Common field that takes advantage of this kind of learning is field of medical imaging, where radiologists go through the entire CT scans and annotate only small subset of slices that usually holds the most important bits of information, producing so called **sparse annotations**. Networks can then use these small subsets to improve their accuracy in comparison with fully unsupervised models.

Chapter 3

Methodology

This chapter serves as a detailed description of methods used for conducting experiments illustrated in chapter 5. It consists of brief information about nature of the datasets used in this work and the process of annotating them, about the training strategy picked for achieving results presented in 5, and about preprocessing pipeline used for shaping the diverse data spectrum into uniform format.

3.1 Prior problems

In addition to segmentation task, I had to deal with unexpected problems that surfaced before I began the development of segmentation method, which lead to possible decrease of overall accuracy of aforementioned method. These problems were:

- Lack of dental datasets
- Lack of labels

Lack of dental datasets

Generally, datasets used for training of neural networks contain thousands upon thousands of training samples. Unfortunately for field of medical imaging, training datasets of such sizes are often unattainable. The reasons behind shortage of training data lies in the amount of time needed for experts to segment labels, unavailability of patient data due to the legal reasons, specificity of segmented objects and possible health implications linked with computed tomography. Usage of smaller sized dataset inclines to overfitting, thus resulting in loss of generalization capability on other than the training data.

Sensitivity of dental records adds up to the already large pool of constraints limiting the number of publicly available training samples. At the time of writing, there still does not exist unrestricted collection of dental volumes [14] and to make matters even worse, the

possibility of extracting acceptable data from other datasets is generally underwhelming, mainly due to the often appearing teeth censoring or unsatisfactory quality of dental regions present in such datasets.

Lack of annotations

Lack of training samples usually leads to even higher lack of annotations for such data.

In attempt to obtain datasets with labels included, I asked various research paper authors [19][4][5][16][39][20][23], if it was possible for them to provide me with some of the data used in their publications. Unfortunately, since every single research group used in-house data in their paper, almost nobody was able to fulfill my request.

3.2 Datasets

This section describes modalities, obtaining process and labeling of different data samples that were used in training datasets.

CT-ORG dataset

CT-ORG dataset [1] is a collection of 140 computed tomography scans acquired from various patients and machines. Every scan comes with its respective set of mutual 3D organ labels, precisely bones and lungs. Some label files are enhanced by additional sets of organ labels, such as bladder, liver, kidneys or brain.

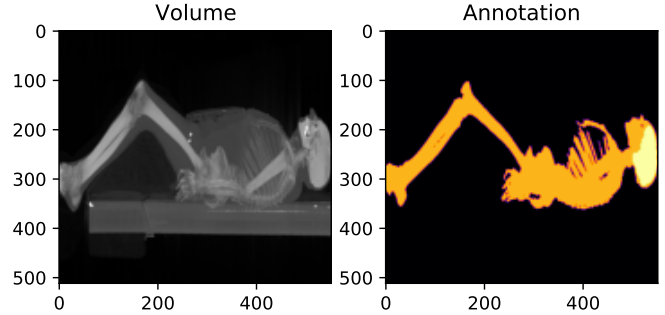


Figure 3.1: Example of volume and its annotations from CT-ORG dataset viewed in sagittal plane

As mentioned above, due to the multiple origins of gathered data, dosage, contrast and even actual content varies from scan to scan. By dosage, images may be classified based on actual number of slices contained in each image as low-dose (usually 50-200 slices) or high-dose (200+ slices). Content variance comes in form of area covered by computed tomography machine, ranging from human torso scans up to scans of entire human body. Every image comes in NifTI format and shares the same constant width and height (512x512) with alternating depth based on the slice count.

The data are split between testing set containing first 21 CT scans and training set remaining 119. For the testing set, bones were manually segmented by experts, where for training set automatic segmentation via morphological image processing was performed.

As the aim with this dataset was to train convolutional neural network for bone segmentation, additional labels were removed via thresholding.

CQ500 dataset

CQ500 is a large collection of head computed tomography scans from 491 different patients. For every patient, usually multiple CT scans with different attributes were provided. Main differences were slice thickness, contrast, dosage and area of focus.

Since purpose of this thesis was to segment only the dental area, I manually reviewed entire dataset and selected only scans that included area of interest based on these conditions:

1. Slice thickness had to be lower than 2mm.
2. Teeth had to be present in at least 25 slices.
3. The number of consecutive slices where dental region was severely distorted by appearance of metallic artifacts could not exceed 5.

Dental CT scans produced by CBCT scanners usually come in high resolution, which leads to volumes with high level of detail in oral region. To simulate that level of detail with CQ500 CT scans, the first and second condition were applied. Last condition was set due to the method which I used to create ground truths of slices affected by metallic artifacts.

After the initial selection resulting in reduction of dataset size from 491 to 42, every volume was manually cropped in Y and Z axis, to reduce the data size as much as possible while preserving the region of interest.

Dental dataset

Dental dataset is combination of 9 CT and CBCT scans, that I have collected from various sources throughout the internet.

Dental dataset			
Source	Num. of scans	File format	Ground truth
Github ¹	3	.png	Yes
Dental CBCT scanner ZCB100 ²	4	.raw	No
Incisix ³	1	.dcm	No
Undisclosed	1	.nii	Yes

Table 3.1: Collection of CT/CBCT data acquired from different sources.

Since data coming from the first source were stored as `.png` images, which meant that they were already processed and scaled to grayscale, therefore resulting in loss of pixel accuracy in representing different types of body tissue. Similar problem occurred with volumes coming from second source, where no header file was present, thus conversion to Hounsfield units or obtaining information about voxel spacing was impossible. To offset this lack of information about `.raw` files, other CBCT scans were used as reference for setting voxel spacings and pixel intensities were scaled to range between 0 and 1, as mentioned in 3.4.

Annotations

To be able to train neural network for aforementioned segmentation task, since I had no prior annotated data, I had to manually segment dental crowns and roots by myself. Annotations were created from selected pieces of dataset CQ500 and for whole Dental dataset.

Annotated dataset	
Source	Index
CQ500	1-42
Incisix	43
Dental CBCT scanner ZCB100	44-47
Undisclosed	48

Segmentation was done with help of open-source dedicated program itk-SNAP¹. For the comparison between results of network trained on sparsely annotated data versus dense annotations, both versions were made.

Table 3.2: Brief overview of training dataset with annotations.

Sparse annotations were created by manually selecting and segmenting one slice per every 16 from original volume, where slices containing either mandible or maxilla bone had extra priority.

In case of **dense annotations**, every fifth slice from volume was annotated manually, while the remaining slices had their annotations interpolated in axial plane from already labeled slices. After that, manual correction of significant interpolation errors was performed. To segment dental tissue that was affected by distortion caused by metallic artifacts, I have used previous or subsequent slices to guess its rough shape.

As a result, dental dataset containing 48 training samples with their respective annotations stored in NifTI format was produced.

¹<https://github.com/kaiwenzha/3D-Teeth-Reconstruction-from-CT-Scans>

²https://figshare.com/articles/dataset/Data1_492_492_303_uchar_raw/3403489

³<https://www.osirix-viewer.com/resources/dicom-image-library/>

¹<http://www.itknap.org/pmwiki/pmwiki.php>

3.3 Training strategy

In pursuit of achieving the best possible segmentation results, the original approach of learning from scratch on target data was extended by addition of augmentations, transfer learning [34] and multi-phase training.

Network architecture

As the choice of neural network architecture best suited for the problem task defined by this thesis, standard **3D U-Net** architecture proposed by Ö. Çiçek et al [42] was chosen, since it generally outperforms any other architectures when it comes to fast and precise volumetric segmentation. For the implementation of 3D U-net, version provided by Z. Zhou et al [41] was used. Since the task of segmenting teeth from other body tissues can be classified as a binary task, **sigmoid** [24] had been utilized as a final activation function and for the network optimisation algorithm, **Adam** [18] optimizer was selected. Regarding the format of input data, network had been trained on **single-channel** 3D images divided into smaller patches of equal shape. The shape of single-channel output from the network was identical to its input, and the each of the pixel values present in output represented its probability of belonging to the tooth area.

Data Augmentation

To prevent the overfitting that tends to appear when training large neural networks such as U-Net used in this thesis on limited training data, utilization of data augmentations [30] had to be taken into consideration. The following data augmentation techniques were applied on the fly during training: random 90° **rotations**, random **flipping**, random **elastic deformations**, random **contrast transformations** and **gamma correction augmentation**.

Multi-phase training

From observations made on training dataset, it is clear that tooth voxels take only small percentage of the entire CT/CBCT volume. That may result in slower convergence of network during training. One of the conventional approaches to this problem is to crop only the tooth region from whole volume. However, this approach might on the other hand result in appearance of false positives in non-tooth regions. To possibly solve both of the aforesaid obstacles, multi-phase training as proposed by S. Lee et al [20] was used. Although instead of three training phases where data consisted of full CBCT volumes, teeth-containing slices and teeth sub-volumes, only two training phases with full volumes and their respective cropped versions were applied.

Transfer learning

As mentioned in 3.3, taking advantage of transfer learning by fine-tuning already pre-trained neural network on different task might improve the accuracy of segmentation results compared to training for segmentation task from beginning. To test this possibility, two different models were prepared.

First model **restTeeth**, had learned to restore distorted dental scans to their original form. As for training dataset for such task, 15 different CBCT scans were used. Entire learning process is further described in section 3.3. Second model **restChest**, had been taken from original publication [41] and serves as indication of how large of an impact would use of model trained on solely small dental dataset have on transfer learning applicability compared to the model trained on substantially larger sample of chest CT scans.

Additionally, to test the extent of transfer learning applicability of model trained for bone segmentation task of full body CT scans on segmentation of dental areas from high resolution CBCT scans, 3D full resolution model **nnUNet** had been arranged. Entirety of model’s preprocessing and training had been handled by self-configuring method developed by F. Isensee, et al [15]. To enhance the final results, cross-validation method on 5 separately trained folds was performed. Training period for each of the respective folds was ~6 days on dual GPU setup.

Models	
Name	Description
nnUNet	U-Net architecture trained on various bones in human body.
restTeeth	Model pretrained for image restoration on small sample of dental CBCT scans.
restChest	Model pretrained for image restoration on large sample of chest CT scans.

Table 3.3: Brief summary of prepared models for testing the applicability of transfer learning on segmentation task.

Image restoration

For the image restoration task, the model was pre-trained in the same way as it was done by authors of the original paper [41]. Contrary to classical method of supervised learning, the pair image - ground truth is replaced by corrupted image serving as network input and its original as ground truth. By applying this principle, model will eventually be able to restore images to the shape similar to their original, while learning important parameters from training samples, such as their appearance, texture, context etc. Even though model trained on image restoration task cannot be applied to other tasks directly, it can be fine-tuned for specific task. These models may then outperform models trained completely from scratch, due to the preserved parameters learned from image restoration.

The image „corruption“ is done by transforming the image by applying various methods, which were also adopted and implemented in similar way to the original publication [41]. These methods are local pixel shuffling, non-linear transformation, out-painting and in-painting. Gradual application of these methods is performed automatically on already pre-processed patches in the similar way as any other type of augmentation. By doing this way, entire training process can become fully self-supervised. Few of the many application of image distortions can be seen in 3.2a - nonlinear transformation and out-painting, 3.2b - local pixel shuffling and in-painting, 3.2c - local pixel shuffling and nonlinear transformation. Above-mentioned methods were executed in following order:

1. To synthetically increase the amount of training data, patches were augmented by applying **mirroring** transformation with 50% chance of execution.
2. First deformation method applied was **local pixel shuffling**. It consists of shuffling the order of pixels present in patch resulting in transformed patch. Similarly to mirroring, local pixel shuffling applied also with 50% chance.
3. Next in line was **nonlinear transformation**. The main principle behind it is to change patch intensities whilst preserving the original intensity distribution. Application rate of nonlinear transformation was 90%.
4. Last data operation was **in-painting** and **out-painting**. It generates arbitrary number of windows of various sizes and places them on top of each other, resulting in a window of complex shape. Then, random pixel intensity value is assigned to the area outside of the window. The general chance for application of both methods is 80%, while in-painting gets applied in 20:80 ratio compared to the out-painting.

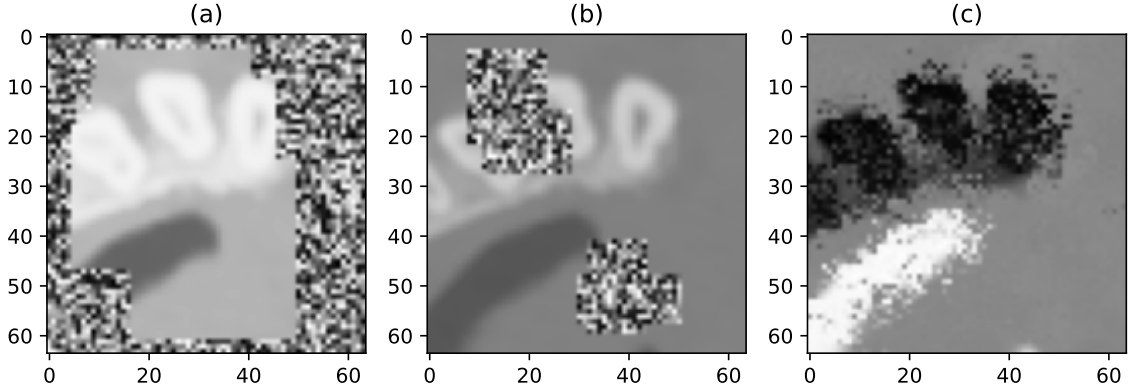


Figure 3.2: Three distinct applications of deformation methods on training images intended for image restoration task

For the process of training the model for image restoration task, mean squared error $L2_{norm}$ and **Adam** optimizer with learning rate 0.001 was chosen.

Similarly to the every training process in this thesis, after each epoch, if model reached new all time low average epoch validation loss, its checkpoint was saved. The training

itself was eliminated after 24h (time limit for multi-GPU training) of training on 2 GPUs and iterated over 400 epochs (1 epoch = entire dataset). The lowest measured average validation loss reached $L2_{norm} = 0.0016$.

3.4 Preprocessing

Before the data were passed through the neural network architecture, they had to be preprocessed, since they came in different file formats (.nii, .dcm, .raw, .png, ...), hence resulting in different pixel intensities, endianness and libraries required for manipulating with such data. Additional reason for preprocessing was to further increase time efficiency and segmentation accuracy of CNN. The whole process can be divided into four following steps:

- Resampling
- Normalization
- Cropping
- Padding

Resampling

To achieve isotropic voxel spacing in CT images and to match the large CBCT volumes with small voxel sizes and smaller CT volumes with larger voxel sizes, entire training dataset was resampled to $0.8mm$ isotropic voxel spacing. In case of sparse annotations, Z-dimension representing depth was excluded from resampling procedure, since it sometimes destroyed annotated slices.

Normalization

Contrary to CT scanners, which generally use the same quantitative scale in Hounsfield units for setting pixel intensities, quantitative scales used in CBCT scanners, most commonly used scanners in field of dentistry do differ depending on the specific manufacturer. To offset these inconsistencies between volumes, firstly I calculated their respective pixel intensities in Hounsfield units (or similar scale provided by manufacturer) x_{hu} as follows:

$$x_{hu} = x_{orig} * slope + intercept.$$

Where x_{orig} represents original intensity of the pixel and $slope + intercept$ are dicom tags provided by manufacturer that specify the linear transformation from pixels stored on the disk to their in memory representation. The difference in memory/disk representation is caused by data being usually stored as unsigned integers to occupy as little disk space as possible while avoiding quantization errors. Since pixels may have large range of values, possibly even negative, this form of transformation is needed.

Since not every volume had their slope/intercept present and some of them were even in completely different ranges not even comparable to Hounsfield units, I had to additionally perform min-max scaling to the range between 0 and 1 with use of formula:

$$x_{scaled} = (x_{hu} - x_{min}) / x_{range}.$$

Where x_{hu} can be substituted for x_{orig} when different quantitative scales such as grayscale are present in volumes. x_{min} is a minimum pixel intensity value in data, almost always representing air and x_{range} is range of values present in volume and can be calculated as $x_{max} - x_{min}$, where x_{max} is the maximum pixel intensity value often being bone or metallic artifact.

After scaling each volume to the same range, entire dataset was clipped to the [0.5, 99.5] percentiles of these intensity values, followed by a z-score normalisation formula:

$$x_{norm} = (x_{scaled} - \mu) / \sigma.$$

Where mean μ and standard deviation σ were calculated from each volume separately based on provided individual annotation masks.

Cropping

Cropping can be defined as act of removal of unwanted outer areas from images. In medical imaging, its common practice to reduce the size of the area outside the scanned body parts as much as possible. In addition to cropping aforesaid areas from medical images, I attempted to create fairly simple method that extracts only the dental region from the entire volume.

The initial step from general cropping process in this thesis starts with finding the slice with most of the tissue present. Its done by calculating pixel intensity means for every slice in volume and subsequently finding the slice with highest value. The idea behind using pixel intensity mean originates from the characteristics of the volumes intended for segmentation.

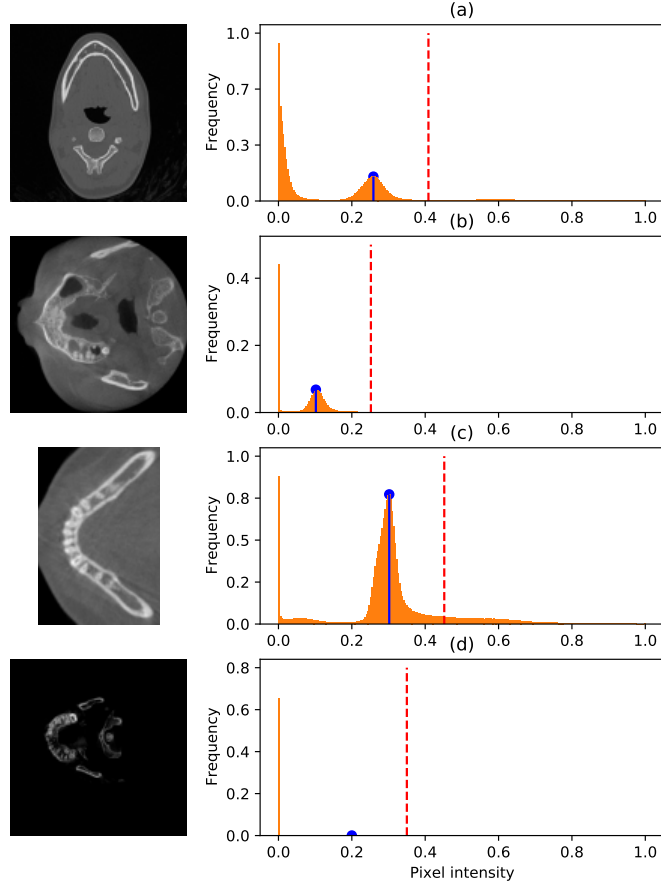


Figure 3.3: Four types of volume pixel intensity histograms with their corresponding slice images (Orange: histogram, Blue: soft tissue peak, Red: computed threshold)

Volumes usually depict human heads or its sections, and pixel intensity value of air in previously scaled data is 0, illustrated as a spike in histogram in 3.3, whilst tissue values are ranging between 0.1-1.0, so the slice with the most tissue will then logically have the least air pixels, thus resulting in highest pixel intensity mean.

Next, 2D array was made marking all pixels where x_{norm} was equal to 0 from the slice with highest mean. This array was used as a mask for calculating the coordinates of a initial cropping window. Since all of the patches have to be same size and I wanted to avoid adding any additional padding as much as possible, the shape of the cropping window was further adjusted to match closest shape that could be divided into equally cut patches.

For extracting only the dental region from entire volume, I first had to compute threshold that separated air, water and soft tissue from bone tissue. Since the volumes had different quantitative scales, threshold could not be a fixed value, but it had to adapt to characteristics of a specific volume. To achieve these thresholds x_{thresh} , I used slightly modified version of formula proposed by S. Lee et al [20], which goes as follows:

$$x_{thresh} = x_{s,peak} + d$$

The soft tissue peak $x_{s,peak}$ was calculated from volume histogram shown in figure 3.3, and constant d represents distance between bone and soft tissue. Even though distance from soft tissue to bone tissue may vary from volume to volume, I observed that $d = 0.15$ generally worked for every volume I used.

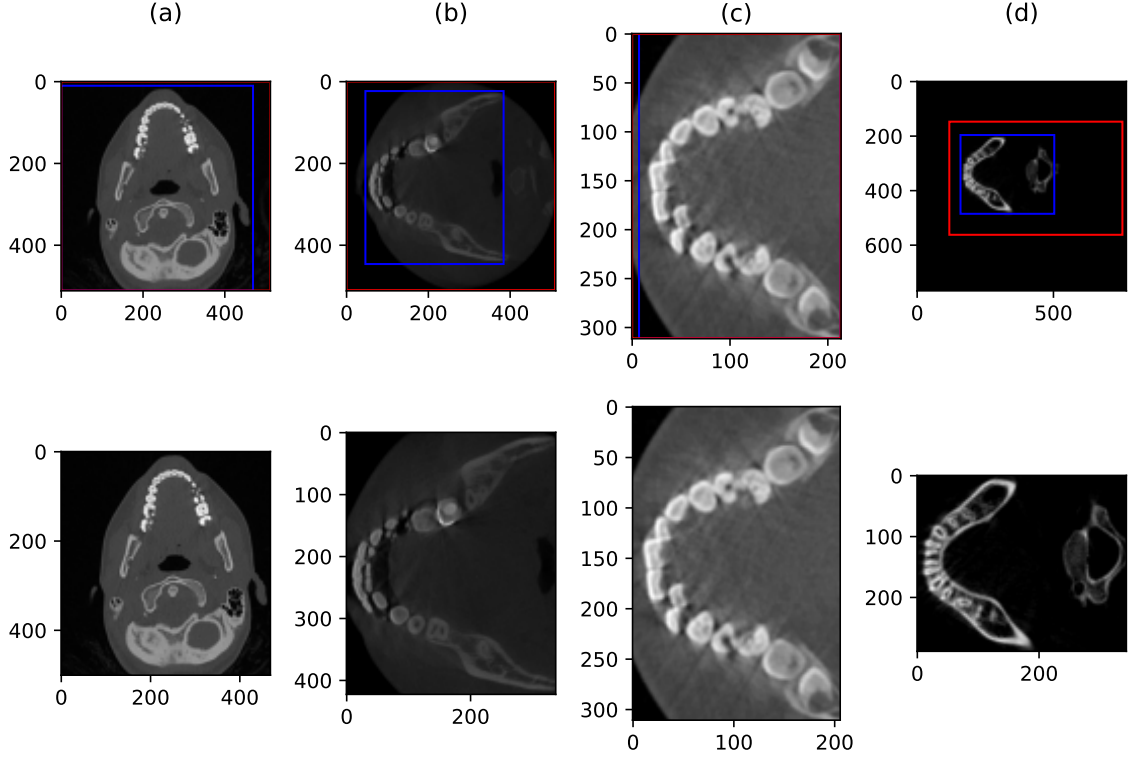


Figure 3.4: Highest mean slices with rectangles marking cropped areas in the first row, cropped images in the second (Red: basic method, Blue: advanced method)

Similarly to basic cropping process used in this thesis, slice with the highest pixel intensity mean had to be found. The difference is that mean was calculated only from bone tissue, rest was ignored. Since pixel intensities of enamel in HU scale are typically higher than intensities of other bones [29], slice with the highest teeth to other bone ratio would be selected.

To create the mask needed for coordinate computation, threshold was again applied to the selected slice, and everything outside bone tissue was set to 0. Rest of the process was identical in both alternatives.

As it is clearly visualized in figure 3.4, even the „advanced“ cropping method managed to incorrectly crop the dental area in volume (a), since the vertebrae mislead the algorithm to arbitrarily inflate the size of bounding rectangle. Although some inconsistencies appeared, the overall dental region detection was solid in most volumes, leading to reducing the patch number needed for coverage of entire volume, thus decreasing the neural network training time.

To prevent any inconsistencies in-between labels and CT data, the same bounding rectangle that I used to crop the CT data was also used to crop volumes containing labels.

Padding

In some cases, it was not possible to make such correction of the cropping window to be able to reach the desired uniformity of patch sizes. I had to artificially alter shape of the volume to achieve desired outcome. Since the further size reduction could lead to damaging the important information stored in volume, volumes were centered and expanded, or „padded“ equally in every direction. As for value used in filling the padded area, two different approaches were chosen:

- Air-fill
- Interpolation

Air-fill is probably the most common approach, where selected area is filled with pixel representation of air, which is in this case 0. This leads to faster pre-processing, but results in unnatural borders in outermost patches, which may affect the segmentation accuracy.

Another option is to estimate the pixel intensity value of newly added pixels based on the already existing ones, therefore creating more natural borders in outermost patches. For interpolating the CT volumes, **third order b-spline interpolation**[38] was applied, while the volumes containing annotations were interpolated by **nearest-neighbour interpolation**[25].

The clear demonstration between these two approaches can be observed in volume (c) in figure 3.5, where interpolated borders in second row appear more natural compared to sharp cuts in the first one.

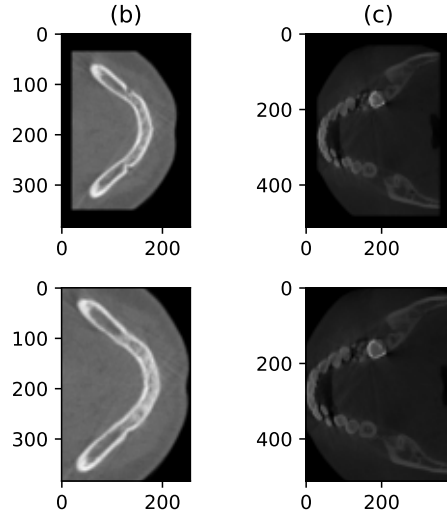


Figure 3.5: Volumes padded with air in the first row, interpolated in the second.

Chapter 4

Implementation

The scripts needed for production of various models described in section 3.3 were implemented in scripting language Python 3.8.5, with support of open source machine learning framework Pytorch¹, version 1.8.1. All models were trained on computing grid Metacentrum, with minimal GPU Cuda compute capability of 6.1. To improve the training speed, both training and validation datasets have been fully loaded to the memory, thus resulting in higher RAM requirements for training.

Implemented scripts can be divided into three distinct categories - training/validation scripts handling the training, fine-tuning and validation process, data manipulation scripts handling preprocessing, postprocessing and dataloading, while the last category consists of „control“ scripts that accept command line parameters and serve as launcher for other mentioned scripts.

4.1 Data manipulation

Preprocessing

Since the data format of CT scan may vary from one to another, they had to be united into one, shared data format before the start of every training cycle. To serve that purpose, class `Preprocessor`, located in `preprocessor.py` was developed. `Preprocessor` accepts scans, preferably in `.dcm`, `.nii` and `.png` formats, but can also handle reading of `.raw` files, with some limitations. It also implements *dataloading*, *resampling* (only for `.nii` images), *normalization*, *cropping* and *padding* operations, as described in section 3.4. With regard to tasks with different learning approaches used in this thesis, `PreprocessorSupervised` and `PreprocessorSelfSupervised` were created as extensions of the original class. The main difference between both classes can be deduced from their respective names, where `PreprocessorSupervised` also processes labels, while the other one does not. After successful pre-processing, all scans are randomly split into training and validation sets with ratio of 3:1 in favor of training set, while the information about split is saved along with

¹<https://pytorch.org/>

the type of supervision and augmentation methods in `settings.json` in the same folder as processed data.

Data-loading

Data-loading before the beginning of every training cycle was handled in same manner as in the preprocessing, with classes `TeethSupervised` and `TeethSelfSupervised` derived from the parent class `Teeth`, with their respective implementations located in the script `dataloader.py`. Information about the validation/training split, augmentations and correct data-loader class is extracted from `settings.json` file prepared beforehand by pre-processor. Full pre-processed volumes are then loaded into the memory and sliced by sliding-window approach for generation of overlapping patches. The **size** of each patch is equal in each dimension with equally large **stride** in every direction.

In case when sparse annotations are used, `TeethSupervised` computes **weight masks**, where zero weight is set for all unannotated slices. Weight masks are then taken into the account when calculating both training and validation losses and evaluation metrics.

Augmentation

Instead of augmenting entire volumes, patch-wise augmentation is applied on the fly, during the training process resulting in relatively unique patch in every iteration. For the specific implementation of each augmentation operations located in `transforms.py`, *mirroring* and *gamma* transformations were adopted from the implementation of nnUNet paper[15] while implementations of *rotation*, *contrast* transformation or *elastic deformation* are taken from pytorch-3dunet² repository created by Adrian Wolny on github. To line up with the patch orientation and dynamic of the training process in used this thesis, slight alterations on the original implementations were performed.

Above-mentioned scripts along with various miscellaneous utility functions from `utils.py` are located in `dataset` subfolder.

4.2 Training and validation

Training

The main component of both validation and training processes is a script `trainer.py` implementing class `UNetTrainer`. This class consists of `fit` function, that accepts `Dataloader`³ objects and iterates over them for exact number of epochs set beforehand by launching script. For each epoch, `fit` calls `train` function that performs batch-wise training over entire course of the training subset and `validate` function that evaluates the current results

²<https://github.com/wolny/pytorch-3dunet>

³<https://pytorch.org/docs/stable/data.html>

of the training process on validation subset. Additionally, these functions send periodical updates with average batch loss and dice to the **Logger**.

Logging

For the purpose of efficient network training progress tracking, especially on remote Metacentrum computing grid, **Logger** was implemented. At the end of each epoch, **Logger** calculates average training/validation loss and dice from the periodical updates received by **UNetTrainer**, along with the epoch time and saves them as an update to both corresponding progress graph and plain text log file located in `pretrained_weights\logs`. Furthermore, it implements the checkpoint saving mechanism, where the current model state is saved in `pretrained_weights` only if average loss measured in the current epoch improves compared to the previously measured best loss. If no advancement is being made in past 100 epochs, it is assumed that there won't be any substantial future network improvements, thus signal is sent to **UNetTrainer** to prematurely terminate the training process.

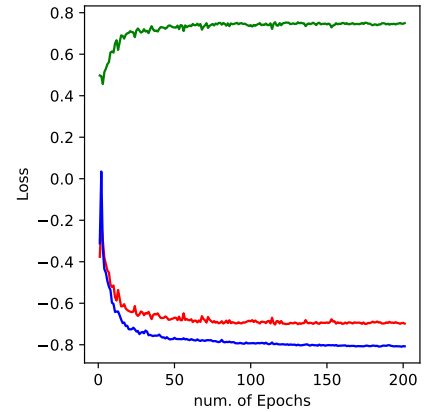


Figure 4.1: Graph tracking the training progress (Blue - training loss, red - validation loss, green - evaluation metric)

Loss and evaluation

Loss function used in this thesis is a combination of *soft dice loss* and *cross entropy*. Implementation of *soft dice loss* was adopted from the authors of nnUNet paper [15] and modified to include weight masks in its calculation. Since the main idea was to train network for binary segmentation task, *binary cross entropy* provided by pytorch library⁴ was sufficient enough. As the evaluation metric, modified version of `DiceCoefficient` by weight mask application originally from pytorch-3dunet repository was used.

4.3 Launching scripts

The main launching scripts, each for its respective task, are `preprocess.py`, `train.py` and `predict.py`. All of them accept different command line parameters, that along with various `.json` files are meant to limit the scale in which scripts have to be modified to prepare different training environments.

⁴<https://pytorch.org/docs/stable/nn.functional.html>

Chapter 5

Experiments

This chapter serves as a summary of all experiments that were carried out on the basis of different approaches to segmentation task described in section 3.3.

As a first step, the appropriate network learning rate and most effective patch size had to be determined. Secondly, various approaches to teeth segmentation were compared between each other on single-phased training including both dense and sparse annotations. Lastly, it was tested if the effects of multi-phased training had any additional impact on quality of segmentation results.

Segmentation results were evaluated by three different metrics, that were collected both throughout and after the training process. First metric used for evaluating the training process was combined loss function that served as main driver for minimizing the segmentation errors produced by network. Along with loss function, average dice score was calculated both for training and validation subset which served as another marker for measuring the network convergence speed or if model was possibly over/underfitting. As a second evaluation metric, again dice score, but this time measured on segmentations provided by final checkpoints of trained models was used on 2 testing CBCT scans, further referenced as CBCT#1 and CBCT#2, that were manually segmented with **dense annotations**, as described in 3.2, along with another scan CBCT#3, segmented by experts that was also used as part of validation subset. Evaluating on testing data provides more objective view for comparing different models, instead of only using evaluations obtained from the validation subset during training process. Third and last evaluation metric is based on visual resemblance between the network outputs and ground truth annotations.

5.1 Determining the best training parameters

The first step towards development of segmentation model was to determine the right patch sizes and most efficient network learning rate. Model for every experiment was trained from scratch, with no prior learned context and training lasted for 200 epochs at maximum.

Since U-net architecture used in this thesis had four 2×2 max-pooling layers, patches had to have shape in multiples of 16. Another constraint proved to be GPU memory, since not many modern GPUs can handle patch sizes above $128 \times 128 \times 128$ without being forced to apply distributed parallelism across multiple GPUs.

Starting from the smallest patch size that captured enough desired context - $32 \times 32 \times 32$, patch size was gradually increased in increments of 32 (except for 96) up until the largest possible size of $128 \times 128 \times 128$. Reasoning behind equal sizes in each dimension originated from the isotropic voxel spacings of resampled data.

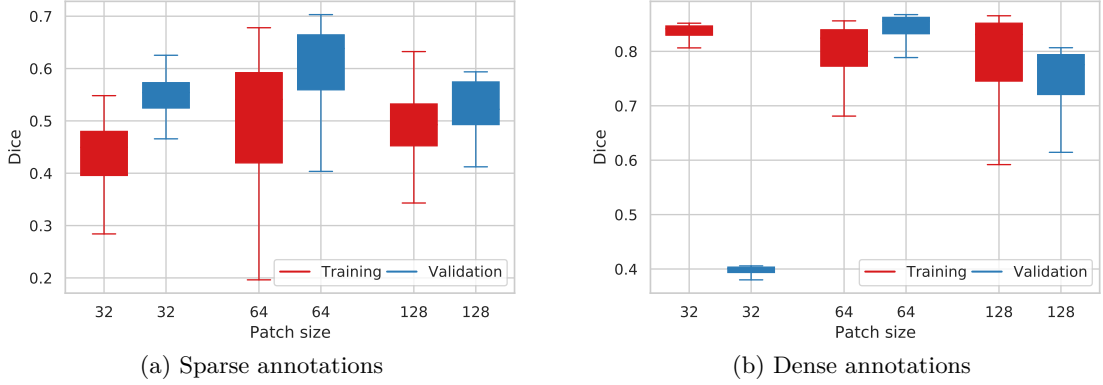


Figure 5.1: Impact of different patch sizes on validation dice score (Light blue - evaluation metric gathered from training samples, Dark blue - evaluation metric calculated from gathered samples)

Graphs portrayed in figure 5.1, show that patch size of $64 \times 64 \times 64$ had proven to be most suitable size for capturing enough context necessary for network learning process, while also limiting the amount of volume reshaping needed for extraction equally sized patches from it. Largest patches $128 \times 128 \times 128$, although probably also capturing enough context, forced preprocessor to perform extensive reshaping via extrapolation due to their large size and relatively low average volume depth (or slice count) in training dataset. That might have caused enough data deformation to decrease the network’s ability to learn, resulting in decrease in segmentation accuracy. On the other hand, patches of size $32 \times 32 \times 32$ failed to encompass enough information for model to be capable of generalization on validation data, as depicted in 5.1b.

To find the learning rate most suitable for training on both sparse and dense annotations, four different learning rates $[10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$ were tested for sparse annotations and three different learning rates $[10^{-2}, 10^{-3}, 10^{-4}]$ were used for training with dense annotations. Experimentation with learning rates started on 10^{-2} in both cases, and learning rate was gradually decreased, up until validation score stopped showing signs of improvement compared to learning rates tested previously.

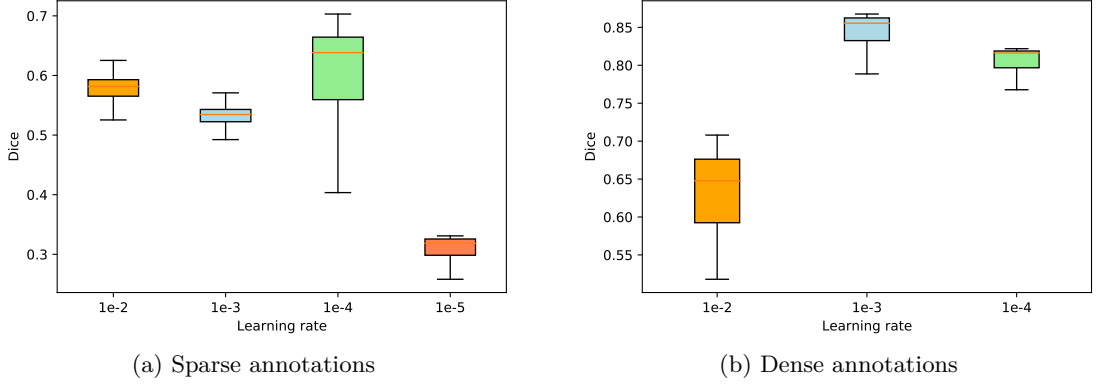


Figure 5.2: Effects of different network learning rates on evaluation metric

Even though experiments in 5.1 had shown that training performance on both sparse and dense annotations was best with the same patch size $64 \times 64 \times 64$, it was not the case for learning rates. As depicted in figure 5.2, which shows performance comparison in-between different learning rates, training on dense annotations performed the best with learning rate of 10^{-3} , while the ability to learn dense volumetric segmentation from sparse annotations had been the highest with learning rate of 10^{-4} . The cause behind this difference might be possibly originating from the limited amount of annotated slices that make the training more vulnerable to undesirable divergent behavior in loss function, so the more careful approach with lower learning rate results in better performance.

After experimenting with various patch sizes and different learning rates, conclusion can be drawn that patch size of $64 \times 64 \times 64$ with learning rates of $10^{-3}/10^{-4}$ each for its respective task suited the training dataset the most.

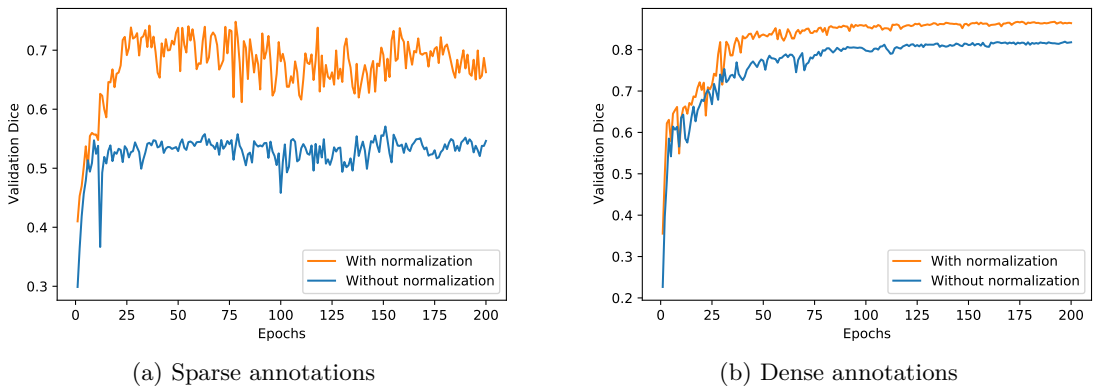


Figure 5.3: Influence of applying z-score normalization on network performance

Figure 5.3 displays effects of normalizing the training dataset on validation dice score for each epoch. It shows its clear improvement compared to the only min-max scaled dataset,

even though z-score normalization caused greater validation dice disparities between each epochs in case of sparse annotations for models trained purely from scratch.

5.2 Single-phase training

The first proper experiment had been the comparison between various different training approaches and baseline segmentation model that was learned from scratch on training dataset. The main idea behind such experiment was to determine if the use of transfer learning had any positive effect on segmentation of dental roots and crowns from CBCT images. To achieve best possible results, learning rates of 10^{-3} for dense and 10^{-4} for sparse annotations were used and patch size was set to $64 \times 64 \times 64$. The training of each segmentation neural network was stopped at 200 epochs, if not sooner due to the lack of improvement persisting for more than 100 epochs. All scans were automatically cropped by preprocessor (`crop` parameter was set to 2).

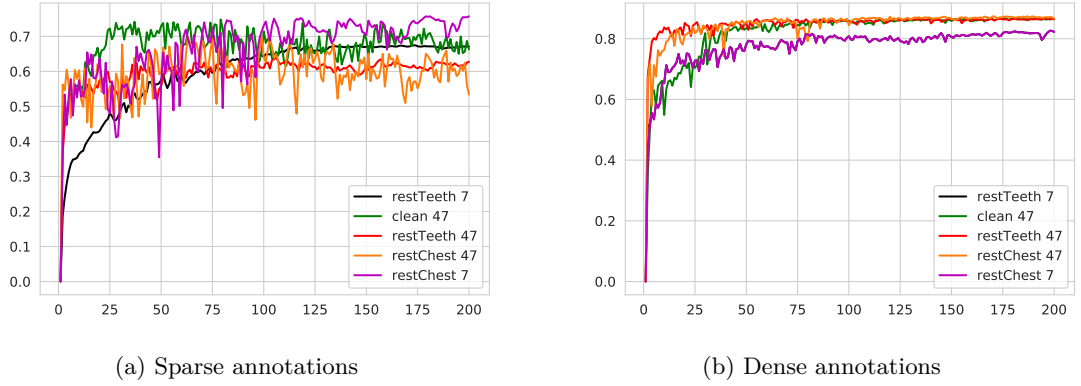


Figure 5.4: Evolution of validation dice score in various models throughout the training process. Numbers behind model names indicate size of training sample

Dense annotations

In case of densely annotated data, six different models were prepared. First, serving as a baseline, was model `clean` trained on full sample of 47 CT/CBCT scans. Models ranging from second to fifth were previously trained for purpose of image restoration task from small sample of 15 dental CBCT scans `restTeeth` and large sample of chest CT scans `restChest`. They were then additionally trained for segmentation task with either full sample of 47 scans, or fine-tuned on limited sample of only 7 CBCT scans. In addition to aforementioned models, last prepared model had been learned to segment various bones in human body. CT-ORG dataset was chosen as training dataset and 3D full resolution self-configuring method `nnUNet` [15] handled both preprocessing and training.

Segmentation results					
Model	Sample size	Accuracy	CBCT#1	CBCT#2	CBCT#3
clean	47	86.74%	87.17%	76.72%	87.74%
restChest	47	87.39%	89.11%	82.66%	89.89%
restChest	7	83.97%	77.21%	70.60%	74.43%
restTeeth	47	86.75%	85.48%	69.64%	83.32%
restTeeth	7	82.58%	86.89%	84.08%	87.47%
nnUNet	140	91.38%	64.83%	5.44%	73.9%

Table 5.1: Table displaying model accuracy on validation subset and dice scores measured on testing CBCT scans segmented by different models with various dense training sample sizes

From the dice scores shown in table 5.1 or from the segmentation results displayed in figure 5.5, conclusion can be drawn that using models such as **restChest** pre-trained on image restoration tasks with large medical datasets as a base for segmentation task yields,

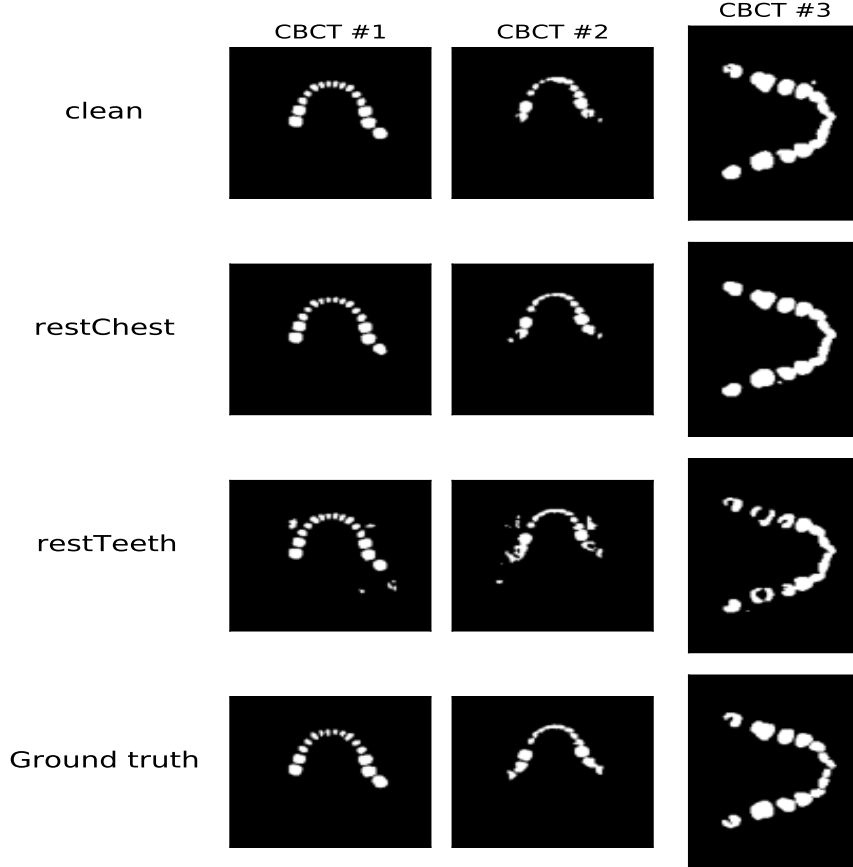


Figure 5.5: Image showing 46-th slice from the CBCT test volumes segmented by models learned with densely annotated data, respectively: clean, restChest trained with 47 scans, restTeeth trained with 7 scans and ground truth labels.

at least in case of CT-dominant dental training datasets, towards slightly better results compared to models such as `clean` trained from scratch. Another benefit of using such pre-trained models instead of randomly initialized models is faster network convergence, as portrayed in figure 5.4.

`restTeeth` trained on full training sample had shown clear decrease in segmentation accuracy compared to the other model pretrained on image restoration task. The cause behind this phenomenon is probably originating from the nature of training dataset, since the model had originally learned to restore images from full CBCT scans, while for segmentation task the amount of CT scans outweighed the CBCT scans present in training set (although the relatively short training time and small training dataset that had been used in pre-training the `restTeeth` model might have had an effect as well). The opposite can be observed in case of fine-tuning the pretrained models with CBCT data, where `restTeeth` heavily outperformed `restChest` and achieved similar results as models trained on full training dataset.

Even though fine-tuning the had shown solid results in terms of dice score, it had produced many false positives in non-dental areas as portrayed in figure 5.5 in both CBCT#1 and CBCT#2.

Suprisingly, `nnUNet` model completely failed to segment CBCT#2 and in case of CBCT#1 and CBCT#3, segmentations included both mandible and maxilla bone.

Sparse annotations

For sparsely annotated scans five separate models were prepared. First model, serving yet again as a baseline was model `clean`, trained on full sample of 47 sparsely annotated CT/CBCT scans. Rest of the models were previously trained for purpose of image restoration task from small sample of 15 dental CBCT scans `restTeeth`, and large sample of chest CT scans `restChest`. They were then additionally trained for segmentation task with either full sample of 47 scans, or fine-tuned on limited sample of only 6 CBCT scans .

Segmentation results					
Model	Sample size	Accuracy	CBCT #1	CBCT #2	CBCT #3
<code>clean</code>	47	70.30%	64.51%	59.45%	74.91%
<code>restChest</code>	47	69.78%	67.86%	72.49%	75.43%
<code>restChest</code>	6	75.71%	65.11%	71.98%	74.00%
<code>restTeeth</code>	47	63.95%	70.25%	67.22%	80.26%
<code>restTeeth</code>	6	67.13%	65.93%	61.26%	78.36%

Table 5.2: Table displaying model accuracy on validation subset dice scores measured on testing CBCT scans segmented by different models with various sparse training sample sizes

In general, all models succeeded in separating teeth from surrounding mandible and maxilla bones, thus already outperforming the `nnUNet` model, but failed to properly recognize the shape of individual teeth or suppress the false positives appearing in spots blind

to the loss function. As shown in figure 5.6, not a single one of presented models succeeded in properly segmenting the frontal teeth from CBCT#2 scan. Closest visual resemblance to the ground truth was achieved by `clean` model, but it also had the highest amount of false positives in non-tooth areas, which had reflected on low dice score in 5.2. Closest numerical resemblance, on the other hand, was measured by `restChest` trained with full training sample.

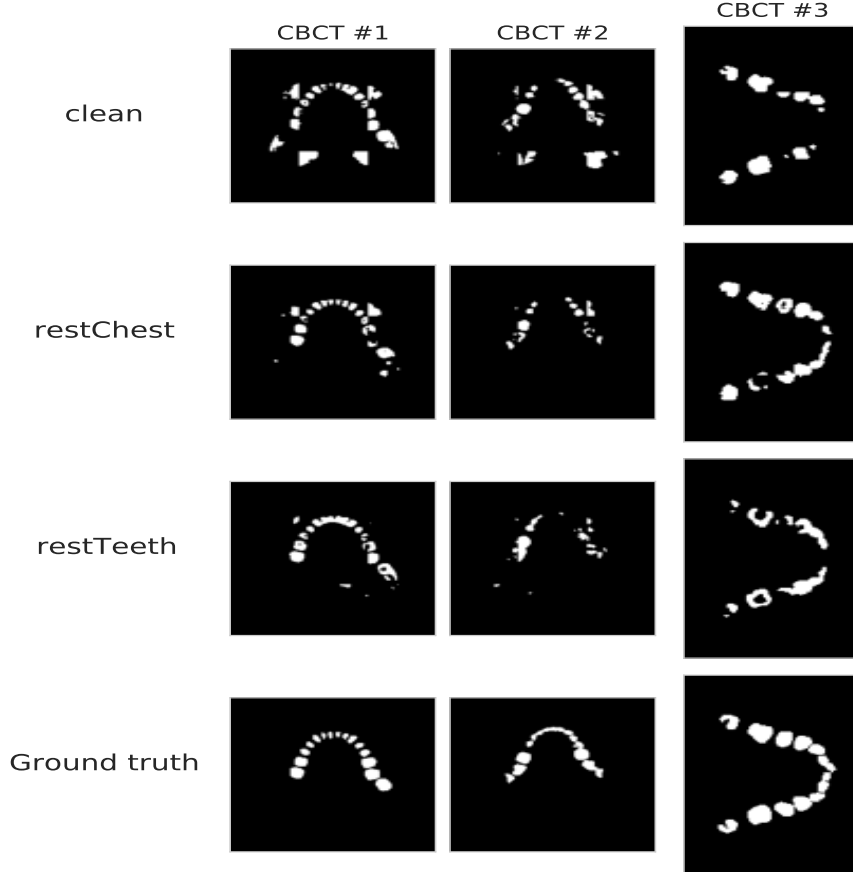


Figure 5.6: Image showing 46-th slice from the CBCT test volumes segmented by models learned on sparsely annotated data, respectively: `clean`, `restChest` trained with 6 scans, `restTeeth` trained with 47 scans and ground truth labels.

5.3 Multi-phase training

Another set of experiments was prepared to determine if multi-phased training had any positive effect on reduction of false positives and overall network performance. Multi-phase training consisted of training on fully sized data without any cropping performed by preprocessor (`crop` parameter was set to 0) in a first phase and second phase, where network learned from automatically cropped data by preprocessor (`crop` parameter was set to 2). To achieve best possible results, learning rates of 10^{-3} for dense and 10^{-4} for sparse annotations were used and patch size was set to $64 \times 64 \times 64$. The training of segmentation

neural network in each phase was stopped at 200 epochs, if not sooner due to the lack of improvement persisting for more than 100 epochs.

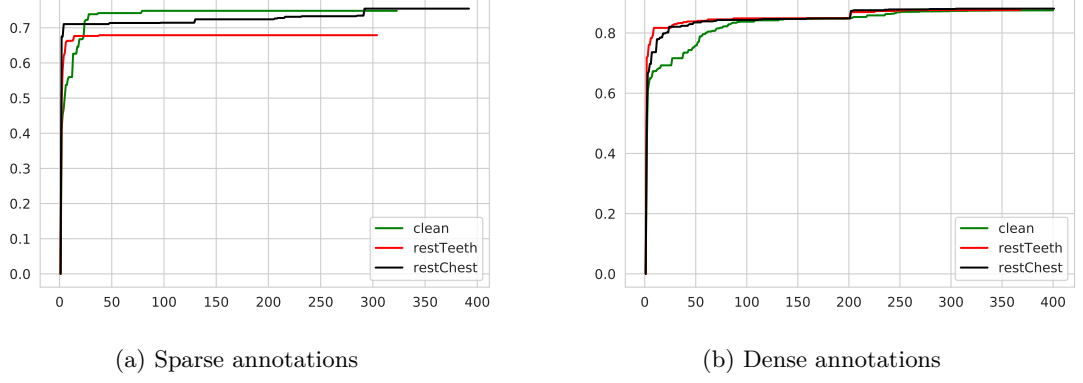


Figure 5.7: Evolution of best validation dice score in various models throughout the multi-phase training process

Since the original representation of all validation dice scores throughout the 400 training epochs resulted in almost unreadable graphs, only the all-time best dice scores that had been measured for each epoch were displayed in figure 5.7. In case of dense annotations, the transition between phases can be clearly noticed by sudden jump in dice score around epoch num. 200, whereas the transitions made by networks trained on sparsely annotated data were much smoother. Even though the transitions differed in both circumstances, they both show a positive trend, thus showing the improvement in network performance with multi-phase training.

For the both sparsely and densely annotated data, only full training sample was used in multi-phase training. To analyze the effects of multi-phase training on overall model accuracy, three separate models were prepared. The model setup was almost identical to the models prepared in single phase experiments 5.2, with omission of fine-tuning.

Segmentation results					
Model	Type	Accuracy	CBCT#1	CBCT#2	CBCT#3
clean	Dense	87.78%	87.09%	85.82%	89.68%
restChest	Dense	88.09%	89.22%	87.06%	91.05%
restTeeth	Dense	87.56%	84.17%	80.60%	87.17%
clean	Sparse	74.63%	75.72%	69.85%	85.36%
restChest	Sparse	73.45%	78.90%	73.56%	82.50%
restTeeth	Sparse	66.50%	68.02%	69.20%	78.56%

Table 5.3: Table displaying model accuracy on validation subset and dice scores measured on testing CBCT scans segmented by different models after passing through the two phases of training

The extension of training process by new phase lead to notable improvements in segmentation accuracy for both types of training data. For the **sparse annotations**, ~10% increase in dice score measured on testing data can be observed, while in case of **densely annotated data** the change appeared to be slightly lower, but still positive nonetheless. These results, along with visual representations in 5.5 indicate that multi-stage training provided new perspective on training data for neural network, allowing it to learn additional information.

As a rebuttal of claim that the longer training time might have been the main driving factor affecting changes in accuracy instead of multi-stage approach, the graph 5.7 tracking the training progress for both types of annotations can be used, where all models failed to improve after 150 epochs up until the deployment of another phase.

Additional phase

As it can be observed in figure 5.7, networks started to struggle in making new breakthroughs towards the end of second phase. To see, if it was possible for models to improve any further, additional phase was added on top of the multi-phase training. The new phase consisted of reducing the training sample from original CT-dominant dataset with 47 scans to only 6 CBCT scans in case of sparse annotated data and 7 CBCT scans for dense annotations, while also decreasing the learning rate by tenfold. All scans were automatically cropped by preprocessor and fine-tuning lasted for additional 200 epochs, if not terminated earlier due to the lack of improvement.

Segmentation results					
Model	Type	Accuracy	CBCT#1	CBCT#2	CBCT#3
clean	Dense	87.78%	87.09%	85.82%	89.68%
restChest	Dense	88.41%	90.23%	87.26%	91.70%
restTeeth	Dense	87.66%	87.87%	80.27%	88.40%
clean	Sparse	76.32%	78.54%	75.64%	85.59%
restChest	Sparse	64.27%	80.17%	76.16%	84.87%
restTeeth	Sparse	67.87%	71.55%	72.43%	84.35%

Table 5.4: Table displaying model accuracy on validation subset and dice scores measured on testing CBCT scans segmented by different models after passing through the three phases of training

The table 5.4 shows that the addition of next phase improves segmentation accuracy of testing samples by ~2% in comparison with only two training phases. Again, the best possible results for both annotation types were achieved by **restChest**, concluding that models with prior knowledge about medical images tend to outperform the approach of learning from zero.

In the end, even the final version of **restChest** trained on **sparsely annotated data** failed to recognize the relatively peculiar tooth shapes appearing in CBCT#2 as portrayed in 5.5. As for the other, more conventional testing samples, results show somewhat solid

tooth segmentation, and persisting failure in distinguishing the non-dental areas in blind spots.

`restChest` trained with **dense annotations** demonstrated comparatively better results to the variant with sparse annotations, where in some cases, such as **CBCT#1**, the output from network had arguably even higher level of precision compared to its ground truth representation.

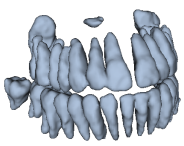
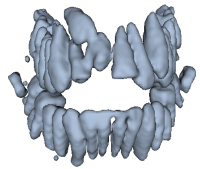
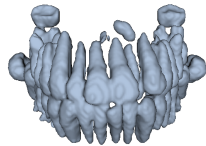
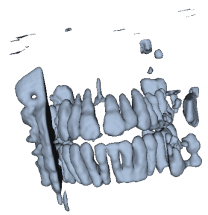
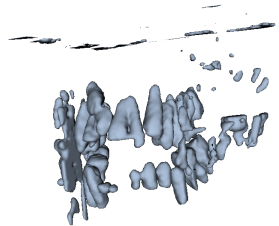
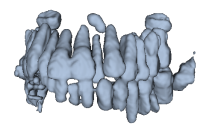
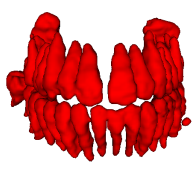
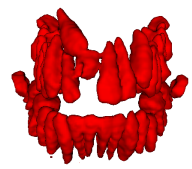
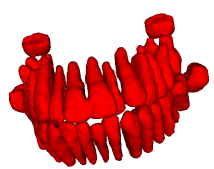
Type	CBCT#1	CBCT#2	CBCT#3
Dense			
Sparse			
Ground truth			

Table 5.5: Final segmentation results by `restChest` model along with ground truth for visual comparison.

Chapter 6

Conclusion

The main goal of this thesis was to deliver method for volumetric segmentation of dental CBCT scans. After evaluation of pros and cons of different segmentation methods, deep learning approach was chosen. All produced models were trained on two segmentation tasks - creating dense segmentations from sparsely annotated data and much more common approach of creating dense segmentations from densely annotated data. In an attempt to enhance the network performance, multi-phase training approach and transfer learning had been applied to both tasks.

Since no such data were publicly available, new dataset consisting of 42 dental areas extracted from CQ500 dataset and 5 CBCT scans was composed. Dataset was then manually annotated with both sparse and dense annotations.

First set of experiments concluded that the best patch size for both of the tasks had been 64x64x64, that the learning rates varied from task to task and that the addition of z-score normalization to dataset showed substantial improvement of model accuracy. Second set of experiments proved that transfer learning is indeed applicable to the both segmentation tasks and in some cases it even outperforms the so called „clean“ models.

Third experiment group had shown the difference between single-phased and two-phased training, with the latter one pulling ahead in terms of overall segmentation dice score. It had also demonstrated that adding an additional phase with decreased learning rate and smaller more precise dataset forced the stagnating models to learn new parameters. The best results for both tasks were produced by their respective restChest models with ~89% average accuracy for task using densely annotated data, and average precision of ~78% for task using sparsely annotated data.

In order to further improve the segmentation results of models trained on sparsely annotated data, multi-channel patches with added posterior probability maps of voxels belonging to the tooth regions instead of single-channel inputs might decrease the amount of false positives, while the larger sample of CBCT scans could perhaps improve the networks ability to recognize various tooth shapes.

Bibliography

- [1] BILIC, P., CHRIST, P. F., VORONTSOV, E., CHLEBUS, G., CHEN, H. et al. The Liver Tumor Segmentation Benchmark (LiTS). *CoRR* [online]. 2019, abs/1901.04056. Available at: <http://arxiv.org/abs/1901.04056>.
- [2] BOYKOV, Y. and KOLMOGOROV, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. 2004, vol. 26, no. 9, p. 1124–1137. DOI: 10.1109/TPAMI.2004.60.
- [3] COMMANDER, C. W. *Maximum cut problem* [online]. Boston, MA: Springer US, 2009. 1991–1999 p. ISBN 978-0-387-74759-0. Available at: https://doi.org/10.1007/978-0-387-74759-0_358.
- [4] CUI, Z., LI, C., CHEN, N., WEI, G., RUNNAN, C. et al. TSegNet: an Efficient and Accurate Tooth Segmentation Network on 3D Dental Model. *Medical Image Analysis* [online]. december 2020, vol. 69, p. 101949. DOI: 10.1016/j.media.2020.101949. Available at: https://www.researchgate.net/publication/347756291_TSegNet_an_Efficient_and_Accurate_Tooth_Segmentation_Network_on_3D_Dental_Model.
- [5] EVAÏN, T., RIPOCHE, X., ATIF, J. and BLOCH, I. Semi-automatic teeth segmentation in Cone-Beam Computed Tomography by graph-cut with statistical shape priors. In: *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)* [online]. 2017, p. 1197–1200. DOI: 10.1109/ISBI.2017.7950731.
- [6] FORD, D. R. and FULKERSON, D. R. *Flows in Networks* [online]. USA: Princeton University Press, 2010. ISBN 0691146675.
- [7] G N, D. S. and SHOBHA, G. Segmentation techniques for target recognition. *WSEAS Transactions on Computers* [online]. january 2008, vol. 7, p. 1555–1563. Available at: https://www.researchgate.net/publication/234804225_Segmentation_techniques_for_target_recognition.
- [8] GAN, Y., XIA, Z., XIONG, J., QUNFEI, Z., HU, Y. et al. Toward accurate tooth segmentation from computed tomography images using a hybrid level set model. *Medical physics* [online]. january 2015, vol. 42, p. 14. DOI: 10.1118/1.4901521. Available at: https://www.researchgate.net/publication/270655804_Toward_accurate_tooth_segmentation_from_computed_tomography_images_using_a_hybrid

- [9] GERON, A. *Hands-on machine learning with Scikit-Learn and TensorFlow : concepts, tools, and techniques to build intelligent systems* [online]. Sebastopol, CA: O'Reilly Media, 2017. ISBN 978-1491962299.
- [10] GOODFELLOW, I., BENGIO, Y. and COURVILLE, A. *Deep Learning* [online]. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [11] HE, K., ZHANG, X., REN, S. and SUN, J. *Deep Residual Learning for Image Recognition* [online]. 2015. Available at: <https://arxiv.org/abs/1512.03385>.
- [12] HIEW, L., ONG, S. and FOONG, K. Tooth segmentation from cone-beam CT using graph cut. *APSIPA ASC 2010 - Asia-Pacific Signal and Information Processing Association Annual Summit* [online]. january 2010, p. 272–275. Available at: https://www.researchgate.net/publication/286378117_Tooth_segmentation_from_cone-beam_CT_using_graph_cut.
- [13] HOCHREITER, S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* [online]. april 1998, vol. 6, p. 107–116. DOI: 10.1142/S0218488598000094. Available at: https://www.researchgate.net/publication/220355039_The_Vanishing_Gradient_Problem_During_Learning_Recurrent_Neural_Nets_and_Problem
- [14] HWANG, J.-J., JUNG, Y.-H., CHO, B.-H. and HEO, M.-S. An overview of deep learning in the field of dentistry. *Imaging science in dentistry* [online]. 2019/03/25th ed. Korean Academy of Oral and Maxillofacial Radiology. Mar 2019, vol. 49, no. 1, p. 1–7. DOI: 10.5624/isd.2019.49.1.1. ISSN 2233-7822. 30941282. Available at: <https://pubmed.ncbi.nlm.nih.gov/30941282>.
- [15] ISENSEE, F., PETERSEN, J., KLEIN, A., ZIMMERER, D., JAEGER, P. F. et al. *NnU-Net: Self-adapting Framework for U-Net-Based Medical Image Segmentation* [online]. 2018. Available at: <https://arxiv.org/abs/1809.10486>
- [16] KAKEHBARAEI, S., SEYEDARABI, H. and TAGHAVI ZENOUEZ, A. Dental Segmentation in Cone-beam Computed Tomography Images Using Watershed and Morphology Operators. *Journal of Medical Signals and Sensors* [online]. april 2018, vol. 8, p. 119–124. DOI: 10.4103/2228-7477.232083. Available at: https://www.researchgate.net/publication/325736041_Dental_Segmentation_in_Cone-beam_Computed_Tomography_Images_Using_Watershed_and_Morphology_Operators.
- [17] KIM, M., YUN, J., CHO, Y., SHIN, K., JANG, R. et al. Deep Learning in Medical Imaging. *Neurospine* [online]. Jun 2020, vol. 17, no. 2, p. 471–472. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6945006/>.
- [18] KINGMA, D. P. and BA, J. *Adam: A Method for Stochastic Optimization* [online]. 2017. Available at: <https://arxiv.org/abs/1412.6980>.
- [19] LEE, J., CHUNG, M., LEE, M. and SHIN, Y. Tooth Instance Segmentation from Cone-Beam CT Images through Point-based Detection and Gaussian Disentanglement. *CoRR* [online]. 2021, abs/2102.01315. Available at: <https://arxiv.org/abs/2102.01315>.

- [20] LEE, S., WOO, S., YU, J., SEO, J., LEE, J. et al. Automated CNN-Based Tooth Segmentation in Cone-Beam CT for Dental Implant Planning. *IEEE Access* [online]. 2020, vol. 8, p. 50507–50518. DOI: 10.1109/ACCESS.2020.2975826.
- [21] LI, Z. and WANG, H. Interactive Tooth Separation from Dental Model Using Segmentation Field. *PLOS ONE* [online]. Public Library of Science. august 2016, vol. 11, no. 8, p. 1–16. DOI: 10.1371/journal.pone.0161159. Available at: <https://doi.org/10.1371/journal.pone.0161159>.
- [22] MCGUINNESS, K. and O’CONNOR, N. E. A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition* [online]. 2010, vol. 43, no. 2, p. 434 – 444. DOI: <https://doi.org/10.1016/j.patcog.2009.03.008>. ISSN 0031-3203. Interactive Imaging and Vision. Available at: <http://www.sciencedirect.com/science/article/pii/S0031320309000818>.
- [23] MINNEMA, J., EIJNATTEN, M. van, HENDRIKSEN, A., LIBERTON, N., PELT, D. et al. Segmentation of dental cone-beam CT scans affected by metal artifacts using a mixed-scale dense convolutional neural network. *Medical Physics* [online]. august 2019, vol. 46. DOI: 10.1002/mp.13793. Available at: https://www.researchgate.net/publication/335475535_Segmentation_of_dental_cone-beam_CT_scans_affected_by_metal_artifacts_using_a_mixed-scale_dense_convolutional_neural_network.
- [24] NWANKPA, C., IJOMAH, W., GACHAGAN, A. and MARSHALL, S. *Activation Functions: Comparison of trends in Practice and Research for Deep Learning* [online]. 2018. Available at: <https://arxiv.org/abs/1811.03378>.
- [25] OLIVIER, R. and HANQIANG, C. Nearest Neighbor Value Interpolation. *International Journal of Advanced Computer Science and Applications* [online]. The Science and Information Organization. 2012, vol. 3, no. 4. DOI: 10.14569/ijacsa.2012.030405. ISSN 2156-5570. Available at: <http://dx.doi.org/10.14569/IJACSA.2012.030405>.
- [26] PARAGIOS, N., CHEN, Y. and FAUGERAS, O. *Handbook of Mathematical Models in Computer Vision* [online]. Springer US, 2005. ISBN 9780387263717. Available at: <https://books.google.sk/books?id=LjCQrqfaCtAC>.
- [27] RONNEBERGER, O., FISCHER, P. and BROX, T. *U-Net: Convolutional Networks for Biomedical Image Segmentation* [online]. 2015. Available at: <https://arxiv.org/abs/1505.04597>.
- [28] ROUHONEN, K. *Graph theory* [online]. 2013 [cit. 2020-12-2]. Available at: http://math.tut.fi/~ruohonen/GT_English.pdf.
- [29] SHETTY, D., URS, A. and KAUR, R. A Radiographic technique for differentiating enamel and dentin in odontogenic tumors. *The Internet Journal of Radiology* [online]. 2009, vol. 12. Available at: <https://www.semanticscholar.org/paper/A-Radiographic-technique-for-differentiating-enamel-Shetty-Urs/feb43135b2e61227dc67d7d6433dc1c3ff037fdb>.
- [30] SHORTEN, C. and KHOSHGOFTAAR, T. M. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data* [online]. Jul 2019, vol. 6, no. 1, p. 60. DOI:

- 10.1186/s40537-019-0197-0. ISSN 2196-1115. Available at:
<https://doi.org/10.1186/s40537-019-0197-0>.
- [31] SIMONYAN, K. and ZISSERMAN, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition* [online]. 2015. Available at:
<https://arxiv.org/abs/1409.1556>.
 - [32] SWIERCZYNSKI, P., PAPIEŻ, B. W., SCHNABEL, J. A. and MACDONALD, C. A level-set approach to joint image segmentation and registration with application to CT lung imaging. *Comput Med Imaging Graph* [online]. april 2018, vol. 65, p. 58–68. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5885990/#bib0075>.
 - [33] SZEGEDY, C., LIU, W., JIA, Y., SERMANET, P., REED, S. et al. Going Deeper with Convolutions. In: *Computer Vision and Pattern Recognition (CVPR)* [online]. 2015. Available at: <http://arxiv.org/abs/1409.4842>.
 - [34] TAN, C., SUN, F., KONG, T., ZHANG, W., YANG, C. et al. *A Survey on Deep Transfer Learning* [online]. 2018. Available at: <https://arxiv.org/abs/1808.01974>.
 - [35] TINGELHOFF, K., EICHHORN, K., WAGNER, I., KUNKEL, M. E., MORAL, A. et al. Analysis of manual segmentation in paranasal CT images. *European archives of oto-rhino-laryngology : official journal of the European Federation of Oto-Rhino-Laryngological Societies (EUFOS) : affiliated with the German Society for Oto-Rhino-Laryngology - Head and Neck Surgery* [online]. march 2008, vol. 265, p. 1061–70. DOI: 10.1007/s00405-008-0594-z. Available at:
https://www.researchgate.net/publication/5597984_Analysis_of_manual_segmentation_in_paranasal_CT_images.
 - [36] WANG, H., ZHOU, Z., LI, Y., CHEN, Z., LU, P. et al. Comparison of machine learning methods for classifying mediastinal lymph node metastasis of non-small cell lung cancer from 18F-FDG PET/CT images. *EJNMMI Res* [online]. Dec 2017, vol. 7, no. 1, p. 11. Available at: <https://arxiv.org/abs/1702.02223>.
 - [37] WANG, S.-C. *Artificial Neural Network* [online]. Boston, MA: Springer US, 2003. 81–100 p. ISBN 978-1-4615-0377-4. Available at:
https://doi.org/10.1007/978-1-4615-0377-4_5.
 - [38] WANG, Z., WANG, K. and AN, S. Cubic B-Spline Interpolation and Realization. In: LIU, C., CHANG, J. and YANG, A., ed. *Information Computing and Applications* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, p. 82–89. ISBN 978-3-642-27503-6. Available at:
https://link.springer.com/chapter/10.1007/978-3-642-27503-6_12.
 - [39] XIA, Z., GAN, Y., CHANG, L., XIONG, J. and QUNFEI, Z. Individual tooth segmentation from CT images scanned with contacts of maxillary and mandible teeth. *Computer Methods and Programs in Biomedicine* [online]. october 2016, vol. 138. DOI: 10.1016/j.cmpb.2016.10.002.
 - [40] YAMASHITA, R., NISHIO, M., DO, R. K. G. and TOGASHI, K. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging* [online]. Aug 2018, vol. 9, no. 4, p. 611–629. DOI: 10.1007/s13244-018-0639-9. ISSN 1869-4101. Available at: <https://doi.org/10.1007/s13244-018-0639-9>.

- [41] ZHOU, Z., SODHA, V., SIDDIQUEE, M. M. R., FENG, R., TAJBAKHS, N. et al. *Models Genesis: Generic Autodidactic Models for 3D Medical Image Analysis* [online]. 2019. Available at: <https://arxiv.org/abs/1908.06912>.
- [42] ÇİÇEK, , ABDULKADIR, A., LIENKAMP, S., BROX, T. and RONNEBERGER, O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. [online]. june 2016. Available at: <https://arxiv.org/abs/1606.06650>.

Appendix A

Contents of the included storage media

- Directory `source` includes all launching scripts along with sub-directories `source/unet` with all training scripts and `source/dataset` where all data manipulation scripts are located.
- Directory `latex_src` contains latex source files for thesis text generation.
- File `requirements.txt` specifying the python packages required for successful run of the scripts included in `source`.
- File `thesis.pdf` contains the text of the bachelor thesis.
- Directory `pretrained_weights` contains all models that were used in the experiments. Best models have `BEST` prefix in their file name.
- File `README.md` containing additional information about the scripts - their command line parameters, limitations etc.

Datasets were not included as a part of the storage media. With the execution of launching scripts, additional directories `preprocessed` and `pretrained_weights/logs` will be generated. Detailed description of the dataset can be found in appendix [B](#).

Appendix B

Dataset

Dataset				
Index	Modality	Slice count	Annotations	Description
01, 02, 05, 06, 08, 10, 14, 19, 21, 22, 25, 30, 35, 39, 41	CT	64	sparse/dense	upper teeth row
03, 04, 13, 15, 16, 17, 18, 23, 36, 42	CT	128	sparse/dense	entire dental region
07, 09, 24, 26, 27, 31, 33, 38, 40,	CT	32	sparse/dense	upper teeth row
11, 12, 20, 28	CT	64	sparse/dense	entire dental region
29	CT	32	sparse/dense	entire dental region
32	CT	32	sparse/dense	upper teeth row with high frequency of metallic artifact appearance
34	CT	32	sparse/dense	upper teeth row with high frequency of metallic artifact appearance
37	CT	128	sparse/dense	entire dental region with high frequency of metallic artifact appearance
43	CT	166	sparse/dense	entire dental region
44	CBCT	244	sparse/dense	entire dental region
45	CBCT	230	sparse/dense	entire dental region
46	CBCT	273	sparse/dense	entire dental region
47	CBCT	303	sparse/dense	entire dental region
48	CBCT	204	dense	entire dental region

Table B.1: Detailed list providing information about slice count, modality, type of annotation and short description of every scan included in training dataset.